ФЕДЕРАЛЬНОЕ АГЕНТСТВО

ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р ИСО 21090— 2016

Информатизация здоровья

ГАРМОНИЗИРОВАННЫЕ ТИПЫ ДАННЫХ ДЛЯ ОБМЕНА ИНФОРМАЦИЕЙ

(ISO 21090:2011, IDT)

Издание официальное

Москва
Стандартинформ
2017

gost.e-ecolog.ru

Предисловие

1 ПОДГОТОВЛЕН на основе официального перевода на русский язык англоязычной версии указанного в пункте 4 стандарта, который выполнен Федеральным государственным учреждением «Центральный научно-исследовательский институт организации и информатизации здравоохранения Министерства здравоохранения Российской Федерации» (ЦНИИОИЗ Минздрава РФ) и Обществом с ограниченной ответственностью «Корпоративные электронные системы»

2 ВНЕСЕН Техническим комитетом по стандартизации ТК 468 «Информатизация здоровья» при ЦНИИОИЗ Минздрава РФ – постоянным представителем ISO/TC 215

3 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Приказом Федерального агентства по техническому регулированию и метрологии от 29 ноября 2016 г. № 1840-ст

4 Настоящий стандарт идентичен международному стандарту ИСО 21090:2011 «Информатизация здоровья. Гармонизированные типы данных для обмена информацией» (ISO 21090:2011 «Health Informatics — Harmonized data types for information interchange», IDT).

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

5 ВВЕДЕН ВПЕРВЫЕ

*Правила применения настоящего стандарта установлены в статье 26 Федерального закона от 29 июня 2015 г. № 162-ФЗ «О стандартизации в Российской Федерации». 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 информационном указателе «Национальные стандарты». В случае пересмотра (замены) или отмены настоящего стандарта соответствующее уведомление будет опубликовано в ежемесячном информационном указателе «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.e-ecolog.ru

© Стандартинформ, 2017

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
3 Термины и определения . . . 2
4 Сокращения . . . 3
5 Соответствие . . . 4
5.1 Общие сведения . . . 4
5.2 Непосредственное соответствие . . . 4
5.3 Косвенное соответствие . . . 6
6 Обзор типов данных . . . 7
6.1 Что такое тип данных? . . . 7
6.2 Определения типов данных . . . 8
6.3 Имена типов данных и повторное использование общераспространенных имен типов данных . . . 8
6.4 Отображение на настоящий стандарт типов данных . . . 9
6.5 Соответствие ИСО/МЭК 11404 . . . 9
6.6 Ссылки на язык UML 2 . . . 9
6.7 Моделирование типов данных . . . 9
7 Типы данных . . . 13
7.1 Общие свойства . . . 13
7.2 Модель верхнего уровня . . . 21
7.3 Базовые типы данных . . . 21
7.4 Текстовые и двоичные типы данных . . . 31
7.5 Кодированные типы данных (терминология) . . . 45
7.6 Типы данных идентификации и местонахождения . . . 58
7.7 Типы данных фамилии, имени, отчества и адреса . . . 68
7.8 Количественные типы данных . . . 90
7.9 Коллекции типов данных . . . 113
7.10 Типы непрерывных множеств . . . 128
7.11 Типы данных, описывающие неопределенность . . . 145
7.12 Структурированный текст . . . 148
Приложение А (обязательное) Представление на языке XML . . . 165
Приложение В (обязательное) Вспомогательные типы данных UML . . . 168
Приложение С (справочное) Отображение на точки зрения БМ-ОРО . . . 171
Приложение D (справочное) Отображение на абстрактные типы данных документа HL7 V3 . . . 172
Приложение Е (справочное) Схема для представления на языке XML . . . 184
Приложение ДА (справочное) Сведения о соответствии ссылочных международных стандартов и документов национальным и межгосударственным стандартам . . . 185
Библиография . . . 186

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Информатизация здоровья

ГАРМОНИЗИРОВАННЫЕ ТИПЫ ДАННЫХ ДЛЯ ОБМЕНА ИНФОРМАЦИЕЙ

Health Informatics. Harmonized data types for information interchange

Дата введения — 2018—01—01

1 Область применения

Настоящий стандарт:

- содержит комплекс определений типов данных, предназначенных для представления и передачи основных понятий, широко используемых при обмене информацией в сфере здравоохранения;
- определяет комплекс специфичных типов данных, пригодных для использования в информационных системах здравоохранения, используемых в различных условиях оказания медицинской помощи;
- определяет семантику этих типов данных с помощью терминологии, нотации и типов данных, определенных в ИСО/МЭК 11404, тем самым расширяя комплекс типов данных, специфицированных в этом стандарте;
- приводит определения этих же типов данных на языке UML, используя терминологию, нотацию и типы данных, определенные в спецификации унифицированного языка моделирования (Unified Modeling Language, UML) версии 2.0;
- приводит представления этих типов данных на языке XML (eXtensible Markup Language).

Требования, определяющие область применения, в основном позаимствованы из HL7 версии 3, ИСО/МЭК 11404, а также из CEN/TS 14796, ИСО 13606 (все части) и предшествующей работы в ИСО над типами данных, используемых в сфере здравоохранения.

Хотя настоящий стандарт может внести практический и полезный вклад в стандартизацию внутренней архитектуры медицинских информационных систем, он в первую очередь предназначен для использования при проектировании внешних интерфейсов или сообщений, обеспечивающих взаимодействие этих систем.

2 Нормативные ссылки

В настоящем стандарте использованы ссылки на следующие документы (для датированных ссылок следует использовать только указанное издание, для недатированных ссылок следует использовать последнее издание указанного документа, включая все поправки).

ISO 4217, Codes for the representation of currencies and funds (Коды для представления валют и фондов)

ISO/IEC 8601 Data elements and interchange formats — Information interchange — Representation of dates and times (Элементы данных и форматы для обмена информацией. Обмен информацией. Представление дат и времени)

ISO/IEC 8824:1990 Information technology — Open Systems Interconnection — Specification of Abstract Syntax Notation One (ASN.1) [Информационные технологии. Взаимодействие открытых систем. Нотация абстрактного синтаксиса версии 1 (ASN.1)][1)]

[1)] Это издание было заменено на серию стандартов ИСО/МЭК 8824:2008 (все части).

ISO/IEC 11404:2007, Information technology — General-Purpose Datatypes (GPD) [Информационные технологии. Типы данных общего назначения (GPD)]

ISO/TS 22220, Health informatics — Identification of subjects of health care (Информатизация здоровья. Идентификация субъектов медицинской помощи)

IETF RFC 1738 — Uniform Resource Locators (URL) (Унифицированные указатели ресурсов)

IETF RFC 1950 — ZLIB Compressed Data Format Specification version 3.3 (Спецификация формата сжатия данных ZLIB версии 3.3)

IETF RFC 1951 — DEFLATE Compressed Data Format Specification version 1.3 (Спецификация формата сжатия данных DEFLATE версии 1.3)

IETF RFC 1952 — GZIP File Format Specification version 4.3 (Спецификация формата сжатия файлов GZIP версии 4.3)

IETF RFC 2045 — Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies [Многоцелевые расширения электронной почты в Интернете (MIME). Часть 1. Формат тела сообщений в Интернете]

IETF RFC 2046 — Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types [Многоцелевые расширения электронной почты в Интернете (MIME). Часть 2. Типы среды]

IETF RFC 2396 — Uniform Resource Identifiers (URI): Generic Syntax [Унифицированные идентификаторы ресурсов (URI). Общий синтаксис]

IETF RFC 2806 — URLs for Telephone Calls (Представление телефонных номеров в форме URL)[1)]

IETF RFC 3066 — Tags for the Identification of Languages (Теги для идентификации языков)

FIPS PUB 180-1 — Secure Hash Standard (Стандарт безопасной хеш-функции)

FIPS PUB 180-2 — Secure Hash Standard (Стандарт безопасной хеш-функции)[2)]

Open Group, CDE 1.1 — Remote Procedure Call specification, Appendix A (Спецификация вызова удаленной процедуры. Приложение А)

HL7 V3 Standard, Data Types — Abstract Specification (R2) (Типы данных. Абстрактная спецификация. Выпуск 2)

Regenstrief Institute, Inc. and the UCUM Organization The Unified Code for Units of Measure (Regenstrief Institute, Inc. and the UCUM Organization Унифицированные коды единиц измерения)[3)]

XML W3C Recommendation, XML Digital Signature (Рекомендации консорциума W3C. Электронная подпись)[4)]

3 Термины и определения

В настоящем стандарте применены следующие термины с соответствующими определениями:

3.1 **атрибут** (attribute): Характеристика объекта, которому присвоены тип и имя.

Примечание — Значение атрибута может меняться в течение жизненного цикла объекта.

3.2 **класс** (class): Дескриптор множества объектов, обладающих аналогичной структурой, поведением и связями.

3.3 **код** (code): Кодированное представление, опубликованное автором системы кодирования в качестве части системы кодирования и являющееся элементом данной системы кодирования.

3.4 **система кодирования** (code system): Управляемая коллекция идентификаторов понятий (обычно кодов), но иногда являющаяся более сложной системой правил и ссылок.

Примечание — Нередко системы кодирования описываются как коллекции уникально идентифицируемых понятий и ассоциированных с ними представлений, отношений и смысловых значений.

Примеры — МКБ-9, LOINC и SNOMED.

[1)] Заменен на документ IETF RFC 3966.

[2)] Редакция документа FIPS PUB 180-1.

[3)] Regenstrief Institute, Inc. and the UCUM Organization, Indianapolis, Indiana, USA. — gost.e-ecolog.ru org/ucum (ссылка от 23 августа 2010 г.).

[4)] World Wide Web Consortium (W3C). — gost.e-ecolog.ru (ссылка от 23 августа2010 г.).

3.5 **понятие** (concept): Унитарное мысленное представление реальной или абстрактной сущности; атомарная единица мышления.

Примечания

1 Понятие должно быть уникальным в данной системе кодирования.

2 Понятие может иметь синонимы в терминах представления и может быть простым или составным термином.

3.6 **соответствие** (conformance): Выполнение указанного требования; соответствие элемента обработки информации требованиям одной или нескольких специфичных спецификаций стандарта.

3.7 **тип данных** (datatype): Множество различных значений, характеризуемое свойствами этих значений и операциями над этими значениями.

3.8 **перечисление** (enumeration): Тип данных, экземпляры которого являются множествами литералов перечисления, задаваемых пользователем.

Примечание — Литералы имеют относительное упорядочение, но никакая алгебра над ними не определена.

3.9 **генерализация** (generalization): Отношение таксономии между более общим классом, интерфейсом или понятием и менее общим классом, интерфейсом или понятием.

Примечания

1 Каждый экземпляр специфичного элемента является также экземпляром общего элемента. Поэтому специфичный элемент обладает всеми свойствами более общего элемента.

2 Более специфичный элемент полностью совместим с более общим элементом и содержит дополнительную информацию.

3 Экземпляр более специфичного элемента может быть использован там, где допустимо использование более общего элемента.

3.10 **элемент обработки информации** (information processing entity): Любая сущность, обрабатывающая информацию и содержащая понятие типа данных, включая другие стандарты, спецификации, средства и службы обработки данных.

3.11 **наследование** (inheritance): Механизм, с помощью которого более специфичные элементы наследуют структуру и поведение более общих элементов.

3.12 **интерфейс** (interface): Спецификация внешне видимой операции класса, не включающая в себя описание внутренней структуры.

3.13 **инвариант** (invariant): Правило о свойстве класса, которое всегда должно быть истинным.

3.14 **операция** (operation): Услуга, которая может быть запрошена у экземпляра класса.

Примечание — Операция имеет имя и список аргументов, которым присвоены имена и типы. Операция возвращает значение определенного типа.

3.15 **специализация** (specialization): Отношение таксономии между более общим классом, интерфейсом или понятием и менее общим классом, интерфейсом или понятием, при котором у более специфичной сущности появляются новые свойства или переопределяются имеющиеся свойства с помощью ограничений, накладываемых на их возможное поведение.

3.16 **набор строковых символов** (string character set): Набор символов, используемых в стандарте для всего строкового содержания.

3.17 **набор значений** (valueSet): Сущность, представляющая уникально идентифицируемое множество допустимых представлений понятий, позволяющая проверить для любого представления понятия, принадлежит оно набору значений или нет.

Примечание — Понятие может быть представлено с помощью отдельного кода или с помощью посткоординируемого сочетания кодов.

4 Сокращения

Для целей настоящего документа применяются следующие сокращения терминов:

CEN — Comité Européen de Normalisation (European Committee for Standardization, a federation of 28 national standards bodies that are also ISO member bodies) (Европейский комитет по стандартизации, федерация из 28 национальных органов стандартизации, которые также являются членами ИСО);

CNE — Coded no exceptions (кодированный без исключений);

CWE — Coded with exceptions (кодированный с исключениями);
GPD — General-Purpose Datatypes (типы данных общего назначения);
HL7 — Health Level 7,
IETF — Internet Engineering Task Force (Инженерный совет Интернета);
OID — Object Identifier (объектный идентификатор);
OMG — Object Management Group (группа управления объектами);
UML — Unified Modelling Language (унифицированный язык моделирования);
W3C — World Wide Web Consortium (консорциум World Wide Web);
XML — eXtensible Markup Language (расширяемый язык разметки).

5 Соответствие

5.1 Общие сведения

Программа, система, элемент либо иная сущность, обрабатывающая информацию, может соответствовать настоящему стандарту либо непосредственно используя описанные в нем типы данных соответствующим образом, либо косвенно с помощью отображения внутренних типов данных, используемых сущностью, на типы данных, описанные в настоящем стандарте.

Примечание — Термин «элемент обработки информации» используется в соответствии с определением 3.10. Аналогичным образом он описан в разделе 4 ИСО/МЭК 11404:2007, а именно: это определение охватывает прикладные программы, а также другие стандарты и спецификации.

5.2 Непосредственное соответствие

5.2.1 Определение непосредственного соответствия

Элемент обработки информации, непосредственно соответствующий данному стандарту, должен:

a) указывать, какие типы данных, указанные в разделе 7, предусмотрены элементом, а какие не предусмотрены;

b) определять пространства значений типов данных, предназначенных для здравоохранения и используемых этим элементом, которые идентичны пространствам значений, описанным в настоящем стандарте;

c) указывать, в какой мере пространства значений типов данных ограничены при использовании в контексте этого элемента;

d) определять операции над типами данных, предназначенных для здравоохранения, отличающиеся от перемещения или преобразования значений, таким образом, чтобы они или выводились из характеристических операций, описанных в настоящем стандарте, или соответствовали им каким-либо иным образом;

e) если для представления этих типов данных в элементе применяется язык XML, то использовать XML-представления, приведенные в настоящем стандарте,

f) публиковать (что необязательно) формальный профиль соответствия, позволяющий точно описать соответствие, или ссылаться на профиль, опубликованный другим элементом обработки информации.

Приведенные выше требования препятствуют использованию спецификатора типа, определенного в настоящем стандарте в целях обозначения любого другого типа данных (однако следует учесть обсуждение области применения имен типов данных, приведенное в 6.2). Другие ограничения на определение элементом обработки информации дополнительных типов данных из них не вытекают. Например, элемент обработки информации, непосредственно соответствующий настоящему стандарту, может продолжать использовать типы данных общего назначения, определенные в ИСО/МЭК 11404, в дополнение к типам данных, предназначенным для здравоохранения.

Из требования d) не следует, что должны быть реализованы все характеристические операции. Могут быть предусмотрены дополнительные операции. Намерение состоит в разрешении таких дополнительных семантических интерпретаций типов данных, которые не конфликтуют с интерпретациями, приведенными в настоящем стандарте. Такие конфликты могут возникнуть только в том случае, когда элемент предоставляет для данного типа такие операции, при которых конкретная характеристическая операция не может быть реализована или не является конструктивной.

Примерами элементов, для которых может быть обеспечено непосредственное соответствие, служат языковые определения или спецификации, выпускаемые для сферы здравоохранения, их на-

тации, а также определения, приведенные в настоящем стандарте. Кроме того, не должен запрещаться буквальный перевод синтаксиса и определений типов данных, приведенных в настоящем стандарте, с помощью программных средств или пакетов прикладных программ.

Элементы обработки информации, объявляющие непосредственное соответствие настоящему стандарту, не обязаны всегда использовать для представления понятий те типы данных, которые определены в настоящем стандарте, хотя бы потому, что из наличия типа данных адреса не следует, что он всегда должен использоваться для представления адреса. Однако типы данных, определенные в настоящем стандарте, обязательно должны использоваться в тех случаях, когда они применяются в контексте интероперабельности.

Элементы обработки информации, объявляющие непосредственное соответствие настоящему стандарту, могут накладывать дополнительные ограничения на домены значений любого типа данных в контексте его использования. В объявлении соответствия должны быть точно описаны применение этих ограничений элементом обработки информации и алгоритм обработки тех значений, которые не соответствуют наложенным ограничениям.

С помощью этих критериев можно оценить совместимость характеристических операций, определенных элементом, соответствующим настоящему стандарту. Если операции имеют то же имя, что и операция, определенная в настоящем стандарте, то они совместимы с ней, если операция, вызванная с теми же самыми параметрами, возвращает тот же самый результат. Она может иметь другие параметры, но либо для них должны быть предусмотрены значения по умолчанию, либо должно быть дано дополнительное определение одноименной операции с другим списком параметров.

В элементах обработки информации, объявляющих непосредственное соответствие настоящему стандарту, не обязательно должны присутствовать все типы данных, определенные в настоящем стандарте, либо определенный тип данных. В них могут использоваться другие термины, например, «структуры данных».

5.2.2 Объявления соответствия

Для элемента обработки информации, объявляющего непосредственное соответствие настоящему стандарту, должно быть составлено объявление соответствия.

Предполагается, что другие органы стандартизации составят объявления соответствия настоящему стандарту как в общем смысле, так и в смысле принятия определенных в нем типов данных в конкретном стандарте. Кроме того, предполагается, что в некоторых странах будут опубликованы профили этих типов данных в качестве справочного или нормативного документа. Наконец, производители и заказчики информационных систем здравоохранения могут найти полезным создание, совместное использование и публикацию таких объявлений соответствия.

Настоящий стандарт не предписывает ни какую-либо определенную форму объявления, ни способ публикации. Однако объявление соответствия должно иметь точные формулировки, формальное представление и предоставляться всем заинтересованным сторонам, имеющим отношение к области применения данного элемента обработки информации.

Наряду с требованиями, что объявления соответствия должны содержать формальные объявления о соответствии требованиям, указанным в пунктах a) — d) 5.2.1, настоящий стандарт дополнительно устанавливает следующие правила о том, что в этих объявлениях должно быть упомянуто, или рекомендуется упомянуть, или может быть выбрано.

Следующие правила обязательны для объявлений непосредственного соответствия:

a) указание используемого набора символов и кодировки; по умолчанию — Unicode (см. 6.7.5);

b) если для ведения истории изменений и аудита данных использован альтернативный способ, то описание его отображения на информацию об изменении и аудите типов данных (см. 7.1.3);

c) пояснение способа указания кратности атрибутов и коллекций (см. 7.1.5);

d) описание применения атрибутов nullFlavor, updateMode и flavorId типа данных ANY (см. подраздел 7.3.3);

e) если использованы типы данных физических величин, приведение точного описания того, как и когда в типе данных QTY применены атрибуты expression, originalText и атрибуты различных неопределенностей;

f) пояснение, какие методы могут быть использованы для альтернативных определений уникальности дискретного множества (см. 7.9.3);

g) если использованы типы данных структурированных документов, описание области применения контекста документов и приведение точного описания того, каким образом разрешаются ссылки внутри этого контекста (см. 7.12);

h) описание, в какой мере приемлем формат XML и указание используемого в нем пространства имен (см. А.1).

Следующие правила рекомендованы для объявлений непосредственного соответствия:

a) определение правил по умолчанию языка (см. 7.4.2.3.7);

b) указание, какие языки могут быть использованы в свойстве QTY.expression (см. 7.8.2.3.1);

c) указание, какие коды могут быть использованы в свойстве QSC.code (см. 7.10.8.3);

d) если использованы типы данных структурированных документов, описание, каким образом обеспечивается версионность в том контексте, где она применяется (см. 7.12.12.2.1).

В объявлениях непосредственного соответствия можно также придерживаться следующих правил:

a) определять дополнительные атрибуты тонкости типов данных или указывать дополнительные уполномоченные органы в определениях атрибутов тонкости (см. 6.7.6);

b) приводить дополнительные описания применения производных данных и интерпретации значения DER свойства nullFlavor (см. 7.1.4);

c) определять, каким образом используется свойство controlInformationRootExtension типа данных HXIT (см. 7.3.2.3.4);

d) пояснять, как выбираются телекоммуникационные и почтовые адреса, предназначенные для конкретных целей (см. 7.6.2.3.2);

e) указывать системы кодирования, используемые для различных компонентов типов данных фамилии, имени, отчества и адреса (см. 7.7.3.6 и 7.7.5.6).

5.3 Косвенное соответствие

5.3.1 Определение косвенного соответствия

Элемент обработки информации, косвенно соответствующий настоящему стандарту, должен:

a) предусматривать отображение своих внутренних типов данных на типы данных, используемые в здравоохранении и соответствующие спецификациям раздела 7;

b) указывать, для каких типов данных, описанных в разделе 7, отображаемый тип является частным случаем, более общим случаем,и для каких типов отображение не предусмотрено;

c) указывать, используется ли представление типов данных в формате XML, описанное в настоящем стандарте, если типы данных представляются в этом формате, или, что не обязательно, предложить в XML-представлении альтернативное пространство имен;

d) публиковать (что необязательно) формальный профиль соответствия, позволяющий точно описать соответствие, или ссылаться на профиль, опубликованный другим элементом обработки информации.

Примерами элементов, для которых может быть обеспечено непосредственное соответствие, служат спецификации, выпускаемые для сферы здравоохранения, спецификации прикладных программ, средств разработки программного обеспечения и другие спецификации интерфейсов, а также многие другие элементы, оперирующие понятием, и существующие для них нотации.

Поскольку настоящий стандарт в большей мере рассчитан на косвенное, а не непосредственное соответствие, то в будущем необходимо предложить стандарты для существующих спецификаций, используемых в сфере здравоохранения.

Элементы обработки информации, объявляющие непосредственное соответствие настоящему стандарту, не обязаны всегда использовать для представления понятий те типы данных, которые определены в настоящем стандарте, хотя бы потому, что из наличия типа данных адреса не следует, что он всегда должен использоваться для представления адреса. Однако типы данных, определенные в настоящем стандарте, обязательно должны использоваться в тех случаях, когда они применяются в контексте интероперабельности.

В элементах обработки информации, объявляющих косвенное соответствие настоящему стандарту, не обязательно должны присутствовать все типы данных, определенные в настоящем стандарте, либо определенный тип данных. В них могут быть использованы другие термины, например, «структуры данных».

5.3.2 Объявления соответствия

Для элемента обработки информации, объявляющего косвенное соответствие настоящему стандарту, должно быть составлено объявление соответствия.

Настоящий стандарт не предписывает ни какую-либо определенную форму объявления, ни способ публикации. Однако объявление соответствия должно точные формулировки, иметь формальное представление и предоставляться всем заинтересованным сторонам, имеющим отношение к области применения данного элемента обработки информации.

Наряду с требованиями, что объявления соответствия должны содержать формальные объявления о соответствии требованиям, указанным в пунктах a) — d) 5.3.1, настоящий стандарт устанавливает следующие дополнительнst правила о том, что в этих объявлениях должно быть упомянуто, или рекомендуется упомянуть, или может быть выбрано:

a) указание используемого набора символов и кодировки; по умолчанию — Unicode (см. 6.7.5);

b) описание, какие применяются отношения эквивалентности и каким способом (см. 7.1.2);

c) поясание способа указания кратности атрибутов и коллекций, если таковое имеет место (см. 7.1.5);

d) если использование типы данных структурированных документов, описание области применения контекста документов и приведение точного описания того, каким образом разрешаются ссылки внутри этого контекста (см. 7.12).

Следующие правила рекомендованы для объявлений косвенного соответствия:

a) определение правил по умолчанию языка (см. 7.4.2.3.7);

b) описание отображения между электронной подписью, определенной в спецификации организации W3C, и альтернативными реализациями, если таковые имеют место (см. 7.4.5.1).

В объявлениях непосредственного соответствия можно также придерживаться следующих правил:

a) определять дополнительные атрибуты тонкости типов данных или указывать дополнительные уполномоченные органы в определениях атрибутов тонкости (см. 6.7.6);

b) приводить дополнительные описания применения производных данных и интерпретации значения DER свойства nullFlavor (см. 7.1.4);

c) определять, каким образом используется свойство controlInformationRootExtension типа данных HXIT (см. 7.3.2.3.4);

d) пояснять, как выбираются телекоммуникационные и почтовые адреса, предназначенные для конкретных целей (см. 7.6.2.3.2);

e) указывать системы кодирования, используемые для различных компонентов типов данных фамилии, имени, отчества и адреса (см. 7.7.3.6 и 7.7.5.6);

f) указывать, какие языки могут использоваться в свойстве QTY.expression (см. 7.8.2.3.1);

g) указывать, какие коды могут использоваться в свойстве QSC.code (см. 7.10.8.3);

h) если используются типы данных структурированных документов, описывать, каким образом обеспечивается версионность в том контексте, где она применяется (см. 7.12.12.2.1).

6 Обзор типов данных

6.1 Что такое тип данных?

В ИСО/МЭК 11404 тип данных определен как множество различных значений, характеризуемое свойствами этих значений и операциями над этими значениями (ИСО/МЭК 11404:2007, подраздел 3.12).

Тип данных образован тремя основными свойствами:

- пространство значений;
- множество свойств;
- множество характеристических операций.

В общем случае определения области применения типа данных вращаются вокруг того или иного из следующих понятий;

- неизменность (свойства типа данных не могут меняться, если только не создан новый экземпляр: у типов данных нет жизненного цикла);
- тождество эквивалентности и идентичности (если два типа данных эквивалентны, то они являются одним и тем же экземпляром);
- связность отдельного понятия (каждый тип данных должен представлять пространство отдельных понятий).

Поскольку применение этих понятий к домену информации, используемой в здравоохранении, и к их влиянию на область применения типов данных представляется предметом перспективы, то критерии отбора типов данных для включения в настоящий стандарт основаны на множестве типов, которое образовалось в результате дебатов с различными участвовавшими органами стандартизации, занимающимися разработкой стандартов информатизации здоровья. Поскольку ожидается, что в стандартах и спецификациях информатизации здоровья будет предусмотрено отображение на типы данных, определенных в настоящем стандарте, то процесс отбора был преднамеренно включающим. При развитии других стандартов может быть принято решение представлять эти типы данных с помощью более сложных структур, но при этом должно быть дано разъяснение, как преобразовать эти структуры в типы данных, определенные в настоящем стандарте.

6.2 Определения типов данных

В настоящем стандарте определен комплекс именованных типов данных. С каждым из них связаны и короткое, и длинное имя. Формальным именем типа данных является короткое имя. Каждый тип данных определен двумя разными способами:

- в терминах языка спецификации типов данных и других типов, определенного в ИСО/МЭК 11404;
- на языке UML, используя примитивные типы данных из пакета, представляющего ядро UML.

Определения в терминах ИСО/МЭК 11404 приведены для обеспечения преемственности между данным стандартом и документами общего назначения ИСО/МЭК 11404, а определения на языке UML предназначены для облегчения реализации типов данных программными средствами. Определения в терминах ИСО/МЭК 11404 являются по своей природе семантическими и абстрактными, а определения на языке UML являются определениями конкретных структур. В настоящем стандарте акцент сделан на определения конкретных структур, поэтому определения на языке UML обладают приоритетом над определениями в терминах ИСО/МЭК 11404, которые приведены в целях преемственности с этим стандартом.

Типы данных, определенные в настоящем стандарте, представляют собой реализацию спецификации абстрактных типов данных HL7 V3 Abstract Data Types (R2). Это означает, что можно реализовать обмен информацией, основанный на определениях этой спецификации, используя типы данных, определенные в настоящем стандарте. В приложении В показано, как эти типы данных соотносятся с типами данных, определенным в спецификации HL7 V3 Abstract Data Types (R2).

Типы данных, определенные в настоящем стандарте, не ограничены свойствами, описанными в спецификации HL7 V3 Abstract Data Types (R2), и для реализации этих типов указанная спецификация не требуется. Приведенные в ней семантические определения могут использоваться разработчиками для лучшего понимания применения этих типов данных.

6.3 Имена типов данных и повторное использование общераспространенных имен типов данных

Некоторые имена этих типов данных имеют чрезвычайное сходство с именами, определенными в других спецификациях. Например, в настоящем стандарте определен вещественный тип данных REAL, в ИСО/МЭК 11404 определен тип Real, в ядре UML тоже есть тип данных с именем Real (см. примечание 1 в В.2.7, где обсуждаются числовые типы с плавающей точкой).

В настоящем стандарте не делаются попытки переопределить или заменить вещественные типы данных, описанные в ИСО/МЭК 11404 или в языке UML. Вместо этого в нем введен новый тип данных, который представляет собой «обертку» соответствующего «примитивного» типа данных, построенную на основе функциональности этого типа и включающую его в общую базовую архитектуру.

Существует много спецификаций, языков и технологий реализации, которые определяют аналогичные типы, используя либо указанные выше «примитивные» типы, либо тип REAL, определенный в настоящем стандарте, но присваивая им различные имена, перефразируя общую тему Real, Float, Decimal или Double.

Настоящий стандарт не использует новые, ранее не употреблявшиеся имена для определенных в нем типов данных, представляющих такие базовые понятия. Вместо этого им присваиваются общераспространенные имена. Опять-таки в настоящем стандарте не делается попытка переопределить или заменить любые другие определения, используя эти имена.

Если имена типов данных, определенных в настоящем стандарте, входят в противоречие с именами, использованными в других спецификациях, то разработчикам рекомендуется использовать некоторую форму присваивания им пространства имен, например, с помощью указания префикса в виде некоторой строковой константы в случае, если среда разработки не поддерживает присваивание типам данных пространства имен.

Такой шаблон «обертки примитивного типа данных» используется для типов данных BL, ST, INT, REAL, SET, LIST и BAG, определенных в настоящем стандарте.

6.4 Отображение на настоящий стандарт типов данных

Подобно ИСО/МЭК 11404, в настоящем стандарте предполагается, что определенные в нем типы данных будут использоваться в других спецификациях. В этих спецификациях должно быть указано, каким образом в реализованы типы данных и свойства, описанные в настоящем стандарте. Эти типы данных могут использовать непосредственно, отображать на другие типы или структуры данных, определенные в разных местах, либо вообще не поддерживаться преобразование данных между спецификациями.

6.5 Соответствие ИСО/МЭК 11404

Настоящий стандарт объявляет соответствие ИСО/МЭК 11404. Хотя настоящий стандарт можно считать поддерживающим все типы данных общего назначения, в действительности в нем использованы только следующие типы данных:

- boolean;
- enumerated;
- characterstring;
- integer;
- Real;
- class;
- set;
- bag;
- sequence;
- octet.

Типы данных, предназначенные для здравоохранения, представляют собой конструкции классов, построенные из этих базовых примитивных типах данных. Типы данных, предназначенные для здравоохранения, частично определены с помощью языка определения типов данных, описанного в ИСО/МЭК 11404. Поскольку в нем отсутствуют отношения генерализации/специализации, то все эти типы данных не могут быть определены на языке, описанном в ИСО/МЭК 11404, поскольку в их определениях эти отношения занимают важное место.

6.6 Ссылки на язык UML 2

В настоящем стандарте типы данных определены, используя язык UML. Все эти типы данных являются специализациями класса Classifier, определенного в этом языке, и определены или построены на следующих типах данных, включенных в ядро Kernel языка UML и определенных в спецификации языка ограничений объектов OCL 2:

- enumeration;
- boolean;
- integer;
- string;
- collection;
- sequence;
- set;
- bag.

6.7 Моделирование типов данных

6.7.1 Общие сведения

В настоящем стандарте отношения между отдельными типами данных описаны с помощью Унияязыка UML версии 2. Этот способ моделирования обеспечивает следующие возможности:

- свойства, общие для группы типов данных, которые могут быть определены однократно;
- реализацию механизма, с помощью которого один специфицированный тип данных может быть заменен на другой.

6.7.2 Определения атрибутов

Если не указано иное, все атрибуты и ассоциации имеют значение по умолчанию nil.

6.7.3 Генерализация/специализация

В настоящем стандарте определен ряд представлений типов данных, предназначенных для здравоохранения, в виде классов. Для этих классов широко используются отношения генерализации/специализации. Эти отношения трактуются обычным образом, и любой экземпляр класса может быть заменен на экземпляр специализации этого класса. Однако в некоторых спецификациях, основанных на настоящем стандарте, могут накладываться дополнительные ограничения на то, какие специализации допустимы в конкретном контексте.

6.7.4 Определения перечислений

В настоящем стандарте определен ряд атрибутов, имеющих перечислимые множества допустимых значений. Каждое значение представляет определенное терминологическое понятие. В терминологиях могут быть отношения генерализации/специализации. В настоящем стандарте перечисления определяются тремя способами:

- список кодов в форме перечислений, определяемых в соответствии с ИСО/МЭК 11404;
- список кодов в форме перечислений, определены в соответствии с языком UML;
- таблица, в которой перечисления определяются в текстовом виде.

Такая таблица имеет четыре столбца: Level (уровень), Code (код), Title (описание) и Definition (определение), описание которых приведено в таблице 1.

Таблица 1 — Структура таблицы перечисления

Вид строки	Описание
Уровень	Уровень понятия в терминологической иерархии. Все понятие с одним и тем же уровнем, не разделенные понятием с большим или меньшим уровнем, являются братскими, а все понятия, которые следуют за понятием с более высоким уровнем, являются потомками этого понятия
Код	Код, представляющий понятие. Он использован в перечислениях для идентификации понятия во всех его представлениях или при обмене данными в соответствии с настоящим стандартом. Коды указаны с отступами, отражающими терминологическую иерархию
Описание	Краткое человекочитаемое описание понятия
Определение	Краткое определение назначения понятия

Иерархия внутри перечисления является важной частью спецификации. Хотя в ИСО/МЭК 11404 и в языке UML перечисления определены как линейные списки, каждый элемент обработки информации, объявляющий о непосредственном или косвенном соответствии настоящему стандарту, должен принимать во внимание отношение иерархии при оценке значения (смысла) перечисления. Кроме того, при определении результата некоторых операций настоящий стандарт время от времени использует отношения, определенные в табличных описаниях перечислений.

За исключением перечисления типов частей адреса AddressPartType, иерархии отражают отношения генерализации/специализации (известного также как категоризация). В этих иерархиях дочерние коды имеют более специальное значение по отношению к родительскому. Перечисление AddressPartType по своей природе является композицией; в нем дочерние коды описывают части понятия, представленного родительским кодом.

Для примера в таблице 2 приведено подмножество определения перечисления NullFlavor (причина пустоты).

Таблица 2 — Перечисление NullFlavor, ОИД[1]: 2.16.840.1.113883.5.1008 (подмножество)

Уровень	Код	Описание	Определение
1	NI	NoInformation	Отсутствует какая бы то ни было информация, которую можно вывести из данного исключительного значения. Это наиболее общее исключительное значение, оно также использовано по умолчанию
2	UNK	unknown	Правильное значение имеется, но оно не известно
3	ASKU	asked but unknown	Информация запрошена, но ответ не получен (например, пациенту задали вопрос, а ответ он не знает)
4	NAV	temporarily unavailable	Информация в данное время недоступна, но может стать доступной позже
3	NASK	not asked	Эта информация не запрашивалась (например, пациенту вопрос не задавался)
2	MSK	masked	Информация об этом элементе имеется, но не предоставлена отправителем по причине безопасности, конфиденциальности и т. д. Для получения доступа к этой информации могут существовать альтернативные механизмы. Предупреждение: хотя детали информации и не раскрываются, но предоставление такой причины пустоты может привести к нарушению конфиденциальности информации. Такая причина пустоты может быть указана в ситуации, когда получателя необходимо информировать о том, что информация существует, не предоставляя саму информацию
2	NA	not applicable	В данном контексте правильного значения не существует (например, последний менструальный период у мужчины)

В этой таблице все коды понятий после NI указаны с отступом и имеют более высокий уровень, т. е. эти понятия являются специализациями понятия с кодом NI. Понятия с кодами ASKU, NAV и MASK являются специализациями понятия с кодом UNK, а понятия с кодами UNK, MSK и NA являются братскими и представляют собой не что иное, как специализации понятия с кодом NI. Соответственно, перечисляемое значение ASKU подразумевает, что может быть использовано также перечисляемое значение UNK.

Если иное не указано, то все перечисления, приведенные в настоящем стандарте, управляются организацией HL7, которая на регулярной основе публикует обновления соответствующих таблиц. Смысл значений, приведенных в настоящем стандарте, неизменен, хотя в будущих редакциях они могут быть запрещены. Когда эти обновленные таблицы публикуются организацией HL7 или ISO, то вновь введенные значения перечислений могут быть предварительно согласованы в партнерском соглашении до выпуска новой редакции настоящего стандарта.

ОИД системы кодирования, присвоенный ей организацией HL7, публикуется для ссылки и для облегчения передачи этих перечисляемых значений в терминологические подсистемы, которые могут использовать ОИД в целях однозначной ссылки на код.

6.7.5 Строковые типы данных и кодирование символов

В настоящем стандарте даются ссылки как на тип данных characterstring, определенный в ИСО/МЭК 11404, так и на тип String, определенный в ядре языка UML. Для целей настоящего стандарта оба этих типа определяют одну и ту же функциональность, а именно: неизменную последовательность известной длины, состоящую из нуля или большего числа логических символов. В тексте этот тип данных далее обозначается просто как String.

Примечания

1 Как ИСО/МЭК 11404, так и ядро языка UML определяют дополнительные характеристические операции, которые могут быть полезны при реализации типа данных и считаться применимыми, но не представляют непосредственного интереса для настоящего стандарта.

[1] ОИД = объектный идентификатор.

2 В настоящем стандарте определен также тип данных ST, представляющий собой оболочку типа данных String и предлагающий дополнительную функциональность того, как понятие неизменной последовательности символов укладывается в общий комплекс типов данных, предназначенный для здравоохранения и оперирующий ассоциированными понятиями неопределенности, ненадежности и соответствия.

Тип данных String содержит последовательность логических символов, отличающуюся от последовательности байтов, кодирующих последовательность логических символов.

Примечание 3 — Разработчики должны аккуратно учитывать различие между этими двумя понятиями при реализации этих типов данных.

По умолчанию тип данных String содержит символы в кодировке Unicode. Рекомендуется, чтобы все элементы обработки информации, объявляющие непосредственное или косвенное соответствие настоящему стандарту, указывали, что набор символов Unicode используется во всех строковых типах данных, взятых из настоящего стандарта.

Однако существует несколько наборов символов, которые недостаточно хорошо отображаются на Unicode. В связи с этим несколько стран или территорий юрисдикции могут предписывать другие наборы символов, отличающиеся от Unicode. В этих случаях рекомендуется, чтобы стандарты и спецификации, объявляющие непосредственное или косвенное соответствие настоящему стандарту, обеспечивали использование наборов символов, отличающихся от Unicode, и содержали явное указание, какие именно наборы символов поддерживаются и как они представлены.

Примечание 4 — Естественно, использование наборов символов, отличающихся от Unicode, приводит к росту затрат на реализацию. Поэтому на территориях юрисдикции, выбирающих использование наборов символов, отличающихся от Unicode, должны полностью учесть это обстоятельство.

Набор символов, применяемый в конкретной среде реализации, будь то Unicode или какой-либо другой, будет именоваться в настоящем стандарте как «строковый набор символов».

Вне зависимости от того, используется в качестве строкового набора символов Unicode или иной набор, при кодировании примитивного строкового типа в определенном наборе символов могут использоваться управляющие байты, изменяющие интерпретацию текста. Предполагается, что любая операция со строковым типом учитывает такую возможность. Но поскольку все операции, описанные в настоящем стандарте, применены к логической последовательности символов, эти вопросы далее не обсуждаются.

6.7.6 Атрибуты тонкости

В дополнение к базовым типам данных настоящий стандарт определяет также ряд атрибутов тонкостей типов данных, которые сами по себе не являются типами данных и не определяются ни как классы UML, ни как типы на языке XML-схем. Вместо этого атрибуты тонкости описывают общие шаблоны ограничений типов данных. Поэтому они не могут быть представлены в виде новых атрибутов, новых кодов, значений по умолчанию или других вновь определяемых элементов. Они могут только описывать правила того, какие существующие свойства класса могут быть использованы и каким образом.

Поскольку атрибуты тонкости типов данных не могут представлять новые свойства или новое значение и сами по себе не существуют как независимые классы, то для правильной обработки информации элементы обработки информации не должны учитывать ее атрибуты тонкости. Поэтому любой элемент, объявляющий непосредственное или косвенное соответствие настоящему стандарту, может определять атрибуты тонкости типов данных или ссылаться на определения атрибутов тонкостей, данные другим уполномоченным органом, при условии выполнения правил именования, принятых в стандарте, а именно:

- имена должны состоять из последовательности допустимых символов, то есть букв, цифр, подчеркиваний и точек; непробельные символы Unicode могут использоваться по усмотрению элемента обработки информации;
- имена должны начинаться с имени типа, от которого они произведены, после которого следуют точка, пространство имен, другая точка, а за ними любые дополнительные допустимые символы;
- пространства имен используются для предотвращения конфликта между именами атрибутов тонкостей, описанных разными источниками; пространство имен должно представлять собой или код страны согласно ИСО 3166-1, или допустимый идентификатор территории применения (realm), определенный организацией HL7, или DNS-имя.

Примеры

TS.CA.BIRTH ***Правила для дат рождения, опубликованные соответствующим уполномоченным органом Канады.***

TS.NPFIT.NHS_NUMBER	*Атрибут тонкости идентификатора пациента (NHS Number flavor, фиксированный корень имени), опубликованный в документации по программе NPFIT в Великобритании.*
ED.AU.KESTRAL.DOCUMENT	*Правила формата документов, допустимого для австралийской фирмы Kestral.*

Для атрибутов тонкостей типов данных, определенных в настоящем стандарте, пространство имен указывать не требуется.

Приложения не должны отвергать экземпляр типа данных на том основании, что в атрибуте flavorId указан идентификатор атрибута тонкости, неизвестный приложению. Приложения могут отвергать экземпляр типа данных, который ссылается на атрибут тонкости, не относящийся к этому типу, но не обязаны это делать.

Атрибуты тонкости, определенные в настоящем стандарте, не обязаны быть реализованными элементами обработки информации, объявляющими непосредственное или косвенное соответствие настоящему стандарту, и использоваться в соответствии с этими элементами.

6.7.7 Примеры

Для большинства типов данных приведены примеры, предназначенные для иллюстрации различных аспектов, связанных с использование этих типов.

Все примеры приведены в виде XML-фрагментов в соответствии с формой, документированной в приложении А. Примеры составлены в предположении, что содержащий их документ или элемент XML использует набор символов UTF-8. Для явного указания типа данных в большинстве примеров использована спецификация типа xsi:type, но это не требуется, если тип данных полностью определен контекстом использования или XML-схемой.

При обсуждении примеров могут даваться ссылки на содержание, опубликованное не ИСО или HL7, а другими разработчиками стандартов. Это содержание не является нормативной частью настоящего стандарта.

7 Типы данных

7.1 Общие свойства

7.1.1 Неизменность

Эти типы данных концептуально неизменны. Для типов данных не вводится понятие жизненного цикла, у них нет операций, позволяющих изменить существующий тип данных. Однако в настоящем стандарте типы данных определены как классы с атрибутами, что позволяет значению типа данных изменяться после его создания.

Хотя это может оказаться полезны для реализации, в намерение разработчиков настоящего стандарта не входило определение типов данных, допускающих изменение семантики. В спецификациях, использующих типы данных, определенные в настоящем стандарте, следует исходить из постулата неизменности типов данных. В частности, разработчики должны усердно предотвращать эффекты совмещения имен, которые могут возникать в случаях, когда свойствам типа данных разрешается изменяться после того, как этот тип данных введен в действие.

7.1.2 Равенство

Существуют два аспекта равенства, а именно: «являются ли эти два значения данных одним и тем же экземпляром?» и «представляют ли эти два значения данных одно и то же семантическое понятие?».

Для отражения этих двух аспектов равенства в языках UML/OCL использовано два свойства. Первое обозначается как «=» и означает, что эти два значения являются одним и тем же экземпляром, а второе обозначается «equals» и означает, что эти типы представляют одно и то же понятие. Для примитивных типов данных языка OCL, например integer, эти два свойства имеют одно и то же значение. Для классов языка UML значения этих свойств могут различаться.

В настоящем стандарте для каждого типа данных приведены критерии равенства. Они определяют вторую форму равенства, а именно представляют ли эти два значения данных одно и то же понятие? Эти определения равенства тщательно сконструированы, чтобы удовлетворять критерию, согласно которому операция равенства должна быть рефлексивной, симметричной и транзитивной:

рефлексивность: «x equals x» должно быть истинно;

симметричность: если «x equals y» истинно, то «y equals x» должно быть истинно;

транзитивность: если «x equals y» истинно и «y equals z» истинно, то «x equals z» должно быть истинно.

Поскольку эта форма равенства носит сугубо семантический характер, то определения равенства могут зависеть от типа данных и установление равенства может оказаться непростым. Элементы обработки информации, объявляющие непосредственное соответствие настоящему стандарту, должны удовлетворять этим определениям равенства. Элементы обработки информации, объявляющие косвенное соответствие настоящему стандарту, должны точно указывать, что определения равенства применимы.

Атрибуты тонкости типов данных не влияют на определение равенства.

7.1.3 История и журнал изменений

Базовый тип данных HXIT определяет свойства, используемые для указания интервала времени validTime, в течение которого значение было, есть или будет действительно, а также для указания идентификации события, ассоциированного с изменениями, известной как контрольная информация.

Свойство validTime используется не для регистрации изменений, а для отслеживания того, когда конкретная система ассоциировала конкретную версию данных с понятием; оно позволяет делать утверждения о периоде времени, в течение которого элемент данных был правильным описанием понятия. Например, во многих странах лицо может изменить свою фамилию при замужестве или иными законными способами, поэтому конкретная фамилия ассоциируется с лицом в течение ограниченного времени. Аналогично, адреса и другая контактная информация лица могут изменяться при его переезде в другое место.

Контрольная информация представляет собой идентификатор некоторого события, произошедшего в информационной системе, ассоциируемый с контрольной информацией об изменении в системе этого элемента данных. Эта информация специфично связана с привязкой в системах значения этого элемента данных к понятию, которое его описывает. Контрольная информация может использоваться для ведения журнала изменений значений при многократной передаче из одной системы в другую. Дополнительные детали см. в 7.3.2.

Различные спецификации, использующие настоящий стандарт, могут предлагать альтернативные способы представления этой истории и состава журнала изменений, особенно второй части, относящейся к журналу изменений. В таких случаях спецификации должны содержать указание, обычно в своем объявлении соответствия, каким образом эта информация обработана и потенциально отображена на свойства, определенные в настоящем стандарте.

Типы данных неизменны — их значение не может измениться. Понятия срока действия и контрольной информации не применимы к типам данных. Поэтому в тех случаях, когда тип данных повторно использована как тип атрибута другого типа данных, инвариантные выражения будут указывать, что атрибуты срока действия и контрольной информации должны быть пустыми.

7.1.4 Пустые значения и атрибут причины пустоты nullFlavor

В описании базового типа ANY введено понятие, называемое nullFlavor (атрибут тонкости, описывающий причину пустоты). Хотя это понятие имеет некоторую связь с пустым значением null, определенным в языках UML и OCL, это не одно и то же, и для правильной реализации настоящего стандарта необходимо понимать связь и различия между этими двумя понятиями.

Любой экземпляр класса, определенного в настоящем стандарте, может быть пустым. Это соответствует понятию null, определенному в языках UML и OCL. Пустой экземпляр представляет собой экземпляр типа OclVoid и соответствует всем типам данных. Он не несет никакой другой информацию кроме той, что факт отсутствует. Для задания ограничений на использование этой формы пустоты в атрибутах типов данных, определенных в настоящем стандарте, использованы операции oclIsDefined и oclIsUndefined языка OCL.

С другой стороны, может быть создан экземпляр класса, и его причине пустоты nullFlavor может быть присвоено одной из значений, входящих в перечисление NullFlavor.

В этом случае значение класса называется исключительным типа nullFlavor. Это означает, что такое значение представляет собой исключение из нормального домена значений этого типа. Но из этого не следует, что экземпляр класса, представляющий исключение, не связан правилами, определенными в настоящем стандарте: он тем не менее должен удовлетворять описанным в стандарте инвариантам. Однако для исключительных значений многие из этих правил отличаются, поскольку эти значения представляют собой семантическое исключение из нормальных данных. Все исключительные значения должны иметь причину пустоты nullFlavor, которая предоставляет более точную информацию о том, почему значение является исключением из правил. Противоположностью исключительного значения является правильное значение, то есть значение, у которого отсутствует причина пустоты nullFlavor.

Поскольку для наличия причины пустоты nullFlavor экземпляр типа данных должен быть создан, его другим атрибутам могут быть присвоены значения. Тем самым этот экземпляр отличается от пустого объекта null, определенного в языках UML/OCL. Этот объект не существует и не может иметь непустые атрибуты. Если причина пустоты nullFlavor присутствует, то другим атрибутам могут быть присвоены значения, однако при этом не требуется, чтобы элемент обработки информации использовал какое-либо из этих значений, за исключением следующих случаев, которые следует правильно понимать в ситуации, когда они применимы:

- значение OTH из перечисления NullFlavor для кодируемого типа данных CD;
- значения NINF и PINF для типа данных CD из перечисления NullFlavor соответственно для нижней и верхней границы интервала;
- значение OTH из перечисления NullFlavor для типа данных идентификатора II, у которого нет корня root, но есть расширение extension.

Перечисление NullFlavor описано в таблице 3.

Таблица 3 — Перечисление NullFlavor, ОИД: 2.16.840.1.113883.5.1008

Уровень	Код	Описание	Определение
1	NI	NoInformation	Отсутствует какая бы то ни было информация, которую можно вывести из данного исключительного значения. Это наиболее общее исключительное значение, оно также используется по умолчанию
2	INV	Invalid	Значение, представленное в экземпляре, не принадлежит ограниченному домену значений переменной
3	OTH	Other	Фактическое значение не является элементом ограниченного домена значений переменной (например, в используемой системе кодирования данное понятие отсутствует)
4	PINF	Positive infinity	Положительная бесконечность чисел
4	NINF	Negative infinity	Отрицательная бесконечность чисел
3	UNC	Unencoded	Попытки правильной кодировки информации не предпринимались, но некодированное исходное значение представлено (обычно в атрибуте originalText)
3	DER	Derived	Фактическое значение может существовать, но оно должно быть произведено из предоставленной информации (обычно выражение задается непосредственно)
2	UNK	Unknown	Правильное значение имеется, но оно неизвестно
3	ASKU	Asked but unknown	Информация запрошена, но ответ не получен (например, пациенту задали вопрос, а ответ он не знает)
4	NAV	Temporarily unavailable	Информация в данное время недоступна, но может стать доступной позже
3	NASK	Not asked	Эта информация не запрашивалась (например, пациенту вопрос не задавался)
3	QS	Sufficient quantity	Конкретная величина неизвестна, но известно, что она ненулевая и не указана, поскольку образована из большого объема вещества. «Добавить 10 мг ингредиента X, 50 мг ингредиента Y и достаточное количество воды до 100 мл». Эта причина пустоты может использоваться для представления количества воды
3	TRC	Trace	Содержание отлично от нуля, но слишком мало, чтобы можно было узнать его количество
2	MSK	Masked	Информация об этом элементе имеется, но не предоставлена отправителем по причине безопасности, конфиденциальности и т. д. Для получения доступа к этой информации могут существовать альтернативные механизмы.

Окончание таблицы 3

Уровень	Код	Описание	Определение
			Предупреждение: хотя детали информации и не раскрываются, но предоставление такой причины пустоты может привести к нарушению конфиденциальности информации. Такая причина пустоты может быть указана в ситуации, когда получателя необходимо информировать о том, что информация существует, не предоставляя саму информацию
2	NA	not applicable	В данном контексте правильного значения не существует (например, последний менструальный период у мужчины)

В стандарте ИСО/МЭК 11404 определено понятие сигнального значения (sentinel value), представляющего собой значение в пространстве значений типа данных, к которому применимы не все характеристические операции этого типа. Хотя между значениями nullFlavors и sentinel существуют определенные концептуальные сходства, экземпляры типа данных ANY, у которых заданы причины пустоты nullFlavor, не являются сигнальными значениями. Характеристические функции к этим экземплярам применимы, однако результатом их применения будет некоторая разновидность (flavor) пустого значения null. Вследствие сходства с поведением значения null, определенного в языке OCL, данное свойство получило имя «nullFlavor».

Если специально не оговорено, все операции, определенные в настоящем стандарте, имеют одинаковое поведение, а именно:

- если операция выполнена над пустым значением null, определенным в языках UML/OCL, то результатом является null;
- если операция выполнена над значением, имеющими причину пустоты nullFlavor, то имеет место следующее:
  - если у операции нет параметров, то она возвращает значение nullFlavor. Обычно этим значением будет NA (не применимо), но в зависимости от семантики причины пустоты nullFlavor могут возвращаться и другие значения. При выполнении операций над значениями, имеющими причину пустоты nullFlavor, следует рассматривать семантический смысл этого атрибута тонкости;
  - если у операции есть параметры и какой-либо из них имеет значение null, определенное в языках UML/OCL, то результатом является null;
  - если в выполнение операции вовлечено значение null или значение, имеющее причину пустоты nullFlavor, то результирующим значением должно быть null или значение, имеющее причину пустоты nullFlavor. Из этого правила могут быть исключения, если семантика типов данных и причины пустоты nullFlavor не требует иного. В настоящем стандарте описано несколько исключений для специфичных операций;
- если операция выполняется над правильным значением:
  - если у операции есть параметры и какой-либо из них имеет значение null, определенное в языках UML/OCL, то результатом является null;
  - если в выполнение операции вовлечено значение null или значение, имеющее причину пустоты nullFlavor, то результирующим значением должно быть null или значение, имеющее причину пустоты nullFlavor. Из этого правила могут быть исключения, если семантика типов данных и причины пустоты nullFlavor не требует иного. В настоящем стандарте описано несколько исключений для специфичных операций;
  - в остальных случаях операция выполняется, как предписано.

Некоторые специфичные операции отклоняются от этих правил. Эти отклонения документируют для каждой такой операции.

Один из особых случаев возникает при сравнении различных значений, имеющих причину пустоты nullFlavor, на предмет равенства. Два значения, у которых причина пустоты nullFlavor имеет значение NA (не применимо), считаются равными. В то же время два значения, у которых причина пустоты имеет значение NINF (отрицательная бесконечность), не равны, равно как и два значения, у которых причина пустоты имеет значение PINF (положительная бесконечность), поскольку фактическое значение неизвестно. Очевидно также, что значение, у которого причина пустоты имеет значение NINF (отрицательная бесконечность), не равно значению, у которого причина пустоты имеет значение PINF. В остальных

вариантах обычно небезопасно возвращать что-либо, кроме значения с причиной пустоты nullFlavor (обычно NI — нет информации).

Значение любого типа, имеющее причину пустоты nullFlavor, равную NI, у которого все другие атрибуты имеют пустое значение null или которое удовлетворяет данному условию рекурсивно, семантически эквивалентно значению null, определенному в языках UML/OCL, и при необходимости эти формы могут быть взаимно преобразованы. Большинство простых атрибутов объявлено, используя типы данных языков UML или OCL, например, строковый тип String, и при необходимости они могут быть преобразованы в соответствующие комплексные эквиваленты, например ST, у которых причина пустоты nullFlavor будет иметь значение NI.

Поскольку атрибут может быть или пустым, или обладать причиной пустоты nullFlavor, многие инварианты приобретают форму (x.oclIsDefined and x.isNotNull) implies {условие}. Для некоторых инвариантов такое правило не является необходимым, поскольку из промежуточного результата null (которому равно значение x.isNotNull, если x равен null) следует, что окончательный результат также null, и такой инвариант благополучно не будет выполнен, но в других случаях должна осуществляться защита от пустого значения null, чтобы во всех случаях можно было получить правильный результат.

Понятие nullFlavor представляет собой общую основу для обработки неполных данных, которые нередко встречаются при сборе, обработке и анализе информации, используемой в здравоохранении. Это понятие может также играть особую роль при объявлении соответствия в спецификациях, использующих такие типы данных.

Не все значения причины пустоты nullFlavor могут использоваться с любыми типами данных. Значения PINF и NINF могут использоваться только со специфичными типами (INT, REAL, PQ и TS). Значение UNC может использоваться только в типах данных, имеющих атрибут исходного текста originalText (CD, QTY, QSET и их специализации). Значения QS и TRC могут использоваться только в типе данных PQ.

Два значения причины пустоты OTH и INV, а также другие их специализации отражают различие между действительным значением и значением, представленным в экземпляре. Некоторые из типов значений могут использоваться в качестве представления значения, которое требует последующего преобразования, генерирующего действительное значение. Например, может быть предоставлено выражение, обеспечивающее генерацию фактического значения, принадлежащего требуемому домену значений экземпляра. Другим примером служит некодированное значение типа CD, у которого задан только атрибут originalText. При значениях причины пустоты INV, DER и UNC существует возможность, что некоторое преобразование, основанное на дополнительной информации или дополнительном знании, может сгенерировать правильное значение. А значение причины пустоты OTH и его специализации представляют собой утверждение, что никакого лучшего значения, скорее всего, не существует. Отсюда возникают вопросы доверия — насколько надежен источник, утверждающий, что лучшей информации нет, насколько надежна обработка, доверяющая такому источнику? Однако эти вопросы не разрешимы. Поэтому различие не должно приниматься как абсолютное, его надо трактовать как заявление о намерении со стороны источника.

Хотя значение причины пустоты INV и его специализации представляют исключительные значения в контексте их использования, оно является исключительным только в параметрах, определенных настоящим стандартом. Фактические значения всегда должны соответствовать правилам, определенным настоящим стандартом.

Значение причины пустоты DER должно использоваться только в том случае, когда из контекста, в котором оно применяется, точно известно, как фактическое значение может быть получено из представленной информации. В настоящем стандарте предусмотрено свойство QTY.expression (см. 7.8.2.3.1). В других элементах обработки информации могут быть сделаны иные описания использования значения DER, включенные в их объявления соответствия.

Примечание — Чаще всего правильным значением причины пустоты будет NI, и на разработчиков не возлагается дополнительная обязанность выбора или сохранения правильного значения причины пустоты. Во всех случаях сомнения, какое значение причины пустоты применимо за пределами тех технических требований, которые точно прописаны в настоящем стандарте в части использования значений NA, INV, DER, OTH, PINF и NINF, разработчики вполне безопасно могут использовать значение NI. В частности, если пользователь не ввел значение в поле ввода процедуры сбора данных или данные отсутствуют по какой-то неизвестной причине, значение NI может оказаться наиболее подходящим.

В таблице 4 приведены примеры выбора значений причин пустоты.

Таблица 4 — Примеры выбора значений причин пустоты

Сценарий	Выбранное значение причины пустоты
Пользователь не ввел значение в поле ввода экранной формы	NI
Источник не сконфигурирован для кодирования текстового значения в требуемой системе кодирования (codeSystem)	UNC
Источник не способен закодировать данное введенное текстовое значение в требуемой системе кодирования, поскольку в ней нет кодов, эквивалентных этому значению	OTH
Пациент в бессознательном состоянии и не может назвать свою фамилию	NAV
Система не поддерживает данный элемент	NI
Дозировка не указана, но предоставлено выражение, с помощью которого система-получатель может вычислить правильную дозировку по весу пациента	INV
У пациента нет адреса — нет постоянного места жительства или бездомный	NA
Отчет о длительности продолжающегося побочного действия с помощью типа данных IVL<TS>	IVL.high = NA, поскольку действие продолжается — понятие даты завершения неприменимо
Отчет о длительности побочного действия с помощью типа данных IVL<TS>, если неизвестно, завершилась ли она	IVL.high = UNK — дата завершения неизвестна
Система-источник возвращает персональные данные пациента, но в соответствии с примененной политикой безопасности/конфиденциальности исключает из них адрес. Следует учитывать, что согласно обычной практике безопасности/конфиденциальности получатель не уведомляется, что по уважительной причине какая-то информация от него закрыта. Однако в некоторых случаях рабочий процесс требует, чтобы пользователю было сообщено о закрытии информации. Значение причины пустоты MSK как раз и рассчитано на эти немногочисленные случаи	MSK

7.1.5 Соответствие

Соответствие в том виде, как оно описано в разделе 5, акцентировано на соответствии элементов обработки информации настоящему стандарту. Но есть и другой аспект, который также нередко называют соответствием, который акцентирован на правилах, применяемых к типам данных элементами обработки информации, например другими стандартами, при использовании этих типов.

Любой элемент обработки информации, использующий эти типы данных, может ограничить их применение правилами, представленными на некотором человеческом языке или на некотором формальном машинно-обрабатываемом языке, например, OCL. Кроме того, в настоящем стандарте описаны два дополнительных средства, с помощью которых можно ограничить возможные значения типов данных, а именно «обязательность» (mandatory) и «кратность» (cardinality).

Любому внешнему атрибуту, присвоенному типу данных, описанному в настоящем стандарте, может быть также присвоен номинальный признак «обязателен» с булевским значением true. Если в контексте использования этому признаку присвоено значение true, то экземпляр типа данных должен иметь непустое допустимое значение, не иметь причины пустоты nullFlavor, соответствовать всем ограничениям, предписанным настоящим стандартом и дополнительным ограничениям на домен значений, описанным в модели. Если этому признаку не присвоено значение true и экземпляр типа данных не удовлетворяет ограничениям, описанным в модели ограничений, то этот экземпляр либо должен быть маркирован, используя некоторую форму причины пустоты nullFlavor (хотя другая информация тем не менее может быть указана), либо должен быть полностью опущен. В последнем случае применяется значение причины пустоты по умолчанию (обычно NI).

В контексте использования может также задаваться кратность атрибута. Кратность состоит из минимального значения, выраженного целым числом, и максимального значения, выраженного целым числом или символом «*». Кратность обычно представляется в виде «[минимальное значение].. [максимальное значение]», например, 0..1 или [1..*]. Смысловые значения кратности у атрибутов, основанных на коллекциях, и других атрибутов отличаются.

Для атрибутов, имеющих тип коллекций (<dtref ref="dt-COLL"/> и его специализации), кратность указывает, сколько элементов может содержать коллекция. Максимальное значение кратности * означает, что число элементов коллекции не ограничено.

Примечания

1 Отсюда не следует, что элементы обработки информации должны оперировать бесконечно большими коллекциями данных, просто в самой спецификации не накладывается никаких ограничений на размер коллекции. Минимальное значение кратности указывает, сколько элементов должно быть в коллекции как минимум.

2 В случае коллекции, имеющей атрибут «обязательна», коллекция должна содержать не менее одного непустого элемента (то есть не равного null и не имеющего какой-либо причины пустоты nullFlavor).

Другим атрибутам могут быть присвоены только значения кратности 0..0, 0..1 или 1..1. Кратность 0 означает, что атрибут не должен быть представлен в экземпляре и неявно имеет значение причины пустоты NI. Кратность 1 означает, что атрибут имеет значение, хотя оно может иметь некоторое значение пустоты, но только в том случае, если этот атрибут не объявлен обязательным.

Обязательный атрибут должен иметь минимальное значение кратности 1 или более.

Примечание — Такое использование кратности несколько отличается от стандартного описания кратности атрибутов в языке UML. Если в этом языке атрибуту присвоены тип DSET(CS) и кратность 2..3, то это означает, что должно быть два или три множества значений типа данных. Эти варианты использования кратности не являются несовместимыми, оба могут использоваться элементами обработки информации, объявляющими непосредственное или косвенное соответствие. Из документации должно точно следовать, какой именно из этих вариантов использован.

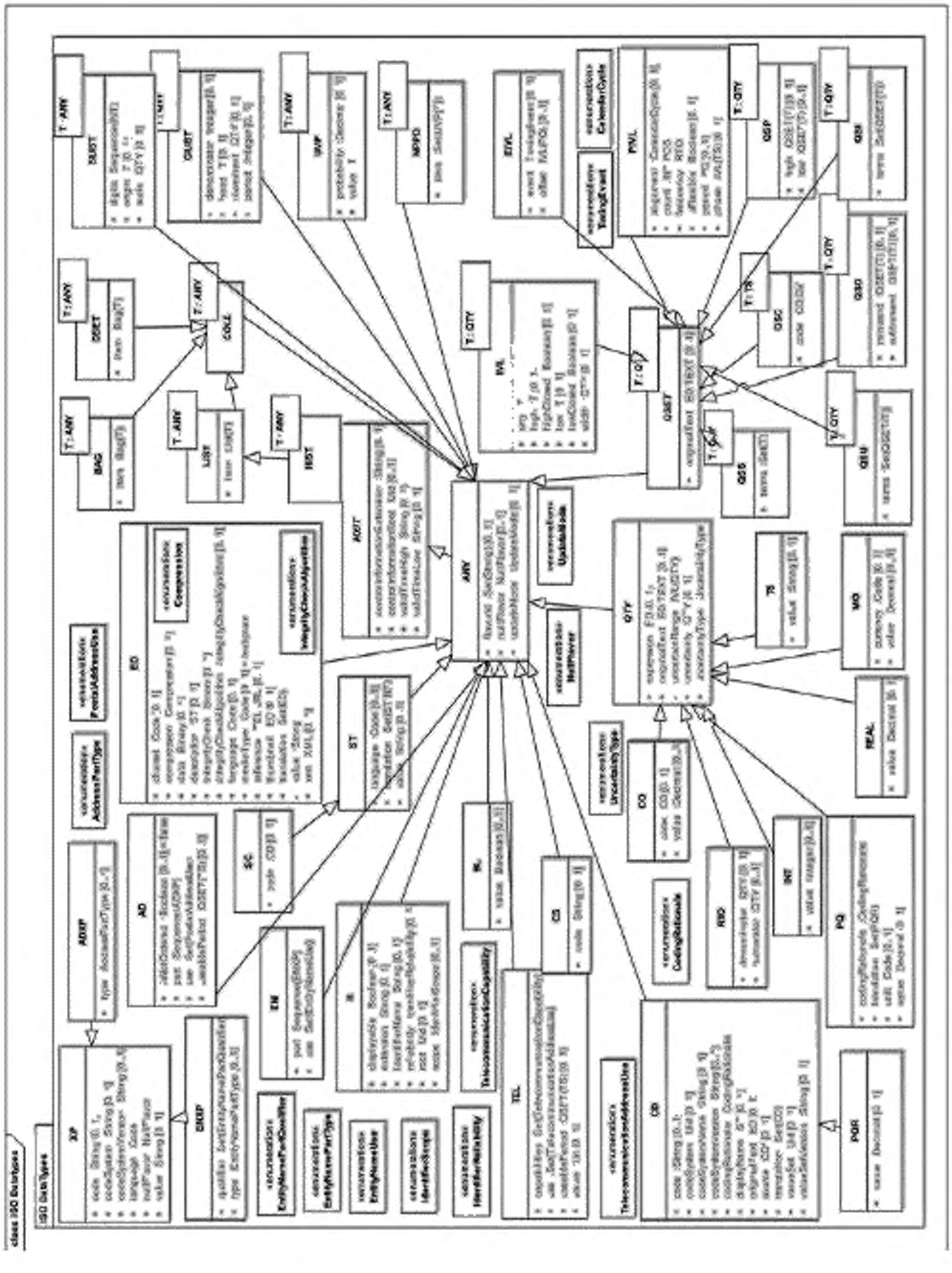


Рисунок 1 — Модель верхнего уровня

7.2 Модель верхнего уровня

Для удобства и последующих ссылок на рисунке 1 в виде диаграммы классов UML приведено обзорное представление типов данных, определенных в настоящем стандарте.

7.3 Базовые типы данных

7.3.1 Введение

Базовые типы данных обеспечивают инфраструктурную поддержку специфичных типов данных, определенных в следующих подразделах (см. рисунок 2).

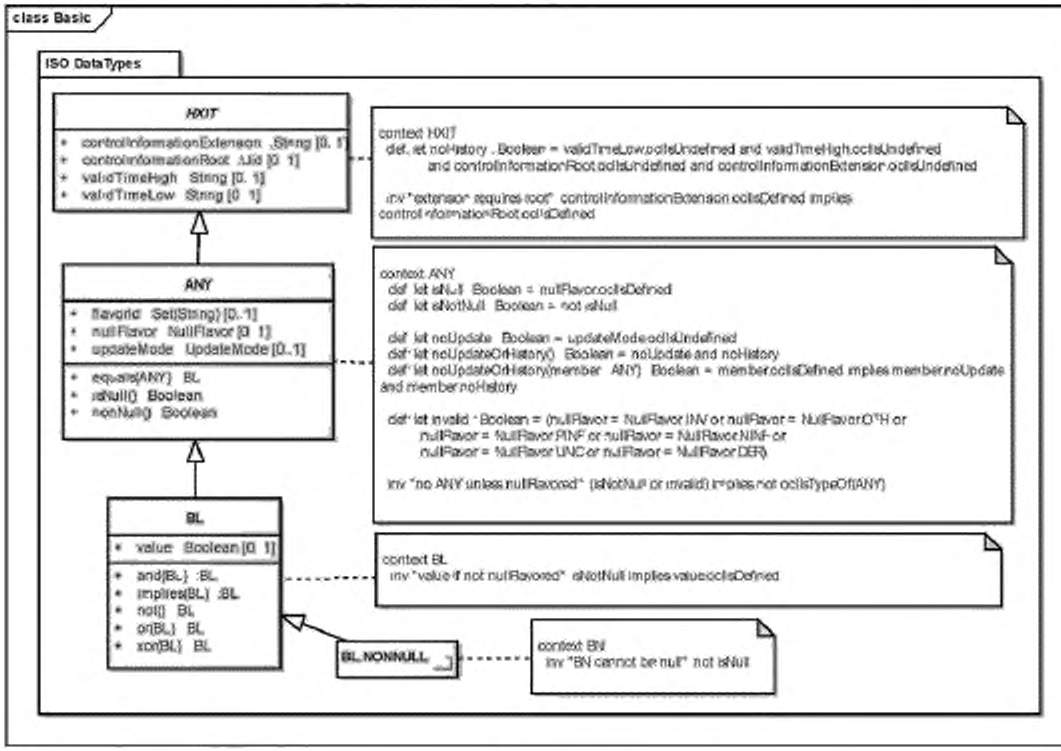


Рисунок 2 — Базовые типы данных

7.3.2 Тип данных HXIT

7.3.2.1 Описание

Абстрактный и частный (private) — этот тип данных не должен быть использован вне типов данных, определенных в настоящем стандарте.

Информация об истории значения: срок действия и ссылка на идентифицируемое событие, которое ввело данное значение в действие.

Вследствие способа определения типов данных ряду их атрибутов присвоены типы данных, произведенные от HXIT. В этих случаях атрибуты HXIT ограничены пустыми значениями. Единственным случаем, когда атрибуты HXIT разрешены в типе данных, служит элемент коллекции (DSET, LIST, BAG, HIST).

В спецификациях, оперирующих этими типами, использование этих атрибутов обычно является предметом дополнительных ограничений.

7.3.2.2 Синтаксис ИСО/МЭК 11404

```
type HXIT = class {
  validTimeLow : characterstring,
```

```
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring
)
```

7.3.2.3 Атрибуты

7.3.2.3.1 validTimeLow: String : момент времени, когда данная информация стала или станет действительной.

Это не момент времени, когда какая-либо система впервые получила данное значение, а то время, когда соответствующая информация стала правильной (например, дата, когда пациент поменял свою фамилию).

7.3.2.3.2 validTimeHigh: String : момент времени, когда данная информация перестала или перестанет быть правильной.

Оба атрибута validTimeLow и validTimeHigh должны быть допустимыми штампами времени, представленными в формате, описанном в 7.8.10.3.1 (значение TS.value).

7.3.2.3.3 controlInformationRoo t: Uid: корень идентификатора события, связанного с присвоением типу данных его указанного значения.

7.3.2.3.4 controlInformationExtension : String: расширение идентификатора события, связанного с присвоением типу данных его указанного значения.

В сочетании корень и расширение идентифицируют конкретную запись о событии реального мира, которая может предоставить дополнительную информацию о значении, например, кто выполнил изменение, почему оно произошло, какая система инициировала изменение. Эта информация существует, поскольку иногда она требуется, но значение представлено во внешнем контексте, не содержащем подходящую связь с контрольной информацией о самом значении. Запись не обязана быть непосредственно доступной или легко доступной. В объявлениях соответствия могут быть приведены дополнительные сведения об этих двух свойствах или о том, как рекомендуется разрешать такую ссылку.

7.3.2.4 Равенство

Поскольку тип данных HXIT является абстрактным и частным, равенство его экземпляров не определено. Атрибуты класса HXIT (validTimeLow, validTimeHigh, controlInformationRoot, controlInformationExtension) никогда не участвуют в определении равенства специализаций этого класса.

7.3.2.5 Инварианты

Если атрибут controlInformationExtension присутствует, то должен присутствовать и атрибут ControlInformationRoot.

Представление инвариантов на языке OCL:

```
def: let noHistory : Boolean =
     validTimeLow.oclIsUndefined and
     validTimeHigh.oclIsUndefined and
     controlInformationRoot.oclIsUndefined and
     controlInformationExtension.oclIsUndefined

inv "extension requires root":
     controlInformationExtension.oclIsDefined implies
     controlInformationRoot.oclIsDefined
```

7.3.2.6 Пример

```
<example
xsi:type="ST" value="Это некоторое содержание"
validTimeLow="200506011000" validTimeHigh="200507031500"
controlInformationRoot="1.2.3.4.5.6">
</example>
```

В этом примере значение атрибута «Это некоторое содержание» было действительно с 10:00 1 июня по 15:00 3 июля 2005 года. Значение «Это некоторое содержание» было присвоено в связи

с событием, идентифицируемым ОИД 1.2.3.4.5.6. Некоторая где-то находящаяся информационная система (определение ее местонахождения должно быть описано в применимом профиле соответствия) способна преобразовать этот ОИД в ссылку, которая может использоваться для идентификации пользователя, осуществившего ввод данного значения в систему.

7.3.3 Тип данных ANY

7.3.3.1 Описание

Специализация типа данных HXIT

Определяет базовые свойства каждого значения данных. Концептуально представляет собой абстрактный тип, то есть никакое правильное значение не может быть только значением данных, не принадлежащим ни к какому конкретному типу. Каждый общедоступный тип данных является специализацией этого общего абстрактного типа данных.

Однако исключительные значения (имеющие причину пустоты nullFlavor) могут иметь тип данных ANY, кроме исключительных значений, у которых подразумевается причина пустоты INV, поскольку для них тип данных должен быть значащим. Следует обратить внимание на то, что не все значения причины пустоты допустимы в типе данных ANY (дополнительные детали см. в 7.1.4).

7.3.3.2 Синтаксис ИСО/МЭК 11404

```
type ANY = class (
  validTimeLow : characterstring,
  validTimeHigh : characterstring,
  controlInformationRoot : characterstring,
  controlInformationExtension : characterstring,
  nullFlavor : NullFlavor,
  updateMode : UpdateMode,
  flavorId : Set(characterstring)
)
```

Правильное использование все трех атрибутов типа данных ANY (дополняющих четыре свойства, унаследованные от типа данных HXIT) связано исключительно со спецификацией, использующей типы данных. Обычно в такой спецификации требуется описать специальные способы управления их использованием. В элементах обработки информации, объявляющих непосредственное или косвенное соответствие, должно быть точно указано, как контролируется использование этих трех атрибутов.

7.3.3.3 Атрибуты

7.3.3.3.1 nullFlavor : NullFlavor: если значение не является правильным, в этом атрибуте указана причина.

Хотя это понятие имеет некоторую связь с пустым значением null, определенным в языках UML и OCL, это не одно и то же, и для правильной реализации настоящего стандарта необходимо понимать связь и различия между этими двумя понятиями. Детальное обсуждение см. в 7.1.4.

Примечание — Понятие причины пустоты nullFlavor включает в себя понятие пустого значения null, определенного в языке UML, а также потенциально полностью заполненные экземпляры, не соответствующие требованиям, предъявляемым к экземпляру (называемые также «исключительными экземплярами»).

Как непустые значения (nonNull), так и значения, имеющие причину пустоты nullFlavor, всегда должны быть действительными в соответствии с правилами, приведенными в настоящем стандарте.

Если этому атрибуту присваивается значение, то оно должно быть взято из системы кодирования HL7 NullFlavor. Текущие значения приведены в таблице 5.

Таблица 5 — Перечисление NullFlavor, ОИД: 2.16.840.1.113883.5.1008

Уровень	Код	Описание	Определение
1	NI	NoInformation	Отсутствует какая бы то ни было информация, которую можно вывести из данного исключительного значения. Это наиболее общее исключительное значение, оно также используется по умолчанию

Окончание таблицы 5

Уровень	Код	Описание	Определение
2	INV	Invalid	Значение, представленное в экземпляре, не принадлежит ограниченному домену значений переменной
3	OTH	Other	Фактическое значение не является элементом ограниченного домена значений переменной (например, в используемой системе кодирования данное понятие отсутствует)
4	PINF	Positive infinity	Положительная бесконечность чисел
4	NINF	Negative infinity	Отрицательная бесконечность чисел
3	UNC	Unencoded	Попытки правильной кодировки информации не предпринимались, но некодированное исходное значение представлено (обычно в атрибуте originalText)
3	DER	Derived	Фактическое значение может существовать, но оно должно быть произведено из предоставленной информации (обычно выражение задается непосредственно)
2	UNK	Unknown	Правильное значение имеется, но оно неизвестно
3	ASKU	Asked but unknown	Информация запрошена, но ответ не получен (например, пациенту задали вопрос, а ответ он не знает)
4	NAV	Temporarily unavailable	Информация в данное время недоступна, но может стать доступной позже
3	NASK	Not asked	Эта информация не запрашивалась (например, пациенту вопрос не задавался)
3	QS	Sufficient quantity	Конкретная величина неизвестна, но известно, что она ненулевая и не указана, поскольку образована из большого объема вещества. «Добавить 10 мг ингредиента X, 50 мг ингредиента Y и достаточное количество воды до 100 мл». Эта причина пустоты может использоваться для представления количества воды
3	TRC	Trace	Содержание отлично от нуля, но слишком мало, чтобы можно было узнать его количество
2	MSK	Masked	Информация об этом элементе имеется, но не предоставлена отправителем по причине безопасности, конфиденциальности и т. д. Для получения доступа к этой информации могут существовать альтернативные механизмы. Предупреждение: хотя детали информации и не раскрываются, но предоставление такой причины пустоты может привести к нарушению конфиденциальности информации. Такая причина пустоты может быть указана в ситуации, когда получателя необходимо информировать о том, что информация существует, не предоставляя саму информацию
2	NA	not applicable	В данном контексте правильное значение не существует (например, последний менструальный период у мужчины)

Синтаксис ИСО/МЭК 11404 атрибута nullFlavor:

```
type NullFlavor = enumerated (NI, INV, OTH, NINF, PINF, UNC, DER, UNK, ASKU,
NAV, QS, NASK, TRC, MSK, NA)
```

При этом, не все значения причины пустоты nullFlavor применимы ко всем типам данных. Причины NINF, PINF, QS и TRC должны использоваться только с типами данных количества QTY. Причина пустоты UNC должна использоваться только с типом данных, у которого есть свойство originalText, и если UNC используется, то свойство originalText должно быть непустым. Если используется причина пустоты DER, то должно непустым выражение expression.

7.3.3.3.2 updateMode : UpdateMode: это свойство позволяет системе-отправителю идентифицировать роль, которую этот атрибут играет в обработке экземпляра, которому он принадлежит.

Если это свойство не пусто, то его значение должно быть взято из системы кодирования HL7 UpdateMode. Текущие значения приведены в таблице 6.

Таблица 6 — Перечисление UpdateMode. ОИД: 2.16.840.1.113883.5.57

Уровень	Код	Описание	Определение
1	A	Add (добавить)	Элемент был (или должен быть) добавлен, если он еще не существует (если он уже существует, то это может трактоваться как ошибка.)
1	D	Remove (удалить)	Элемент был (или должен быть) удален (иногда он только помечается как удаленный). Если элемент является частью коллекции, то удаляются все совпадающие с ним элементы
1	R	replace (заменить)	Элемент уже существовал и был (или будет) пересмотрен (если в момент удаления элемент не существует, то это может трактоваться как ошибка)
1	AR	Add or Replace (добавить или заменить)	Элемент был (или должен быть) добавлен или заменен. О его предшествующем существовании никаких предположений не делается
	N	No Change (не изменять)	Не было (или не должно быть) изменения элемента. Этот код в основном используется, если данный элемент не изменен, но другие атрибуты экземпляра изменились
1	U	Unknown (неизвестно)	Не указано, произошло ли изменение элемента и если произошло, какого вида, или элемент присутствует в форме ссылки или идентифицирующего свойства
1	K	Key (ключ)	Этот элемент является частью информации, идентифицирующей содержащий его объект

Синтаксис ИСО/МЭК 11404 атрибута updateMode:

```
type UpdateMode = enumerated (A, D, R, AR, N, U, K)
```

Если атрибут updateMode отсутствует, то нет сведений о том, как вновь предоставленная информация изменяет уже существующую. В приведенном выше описании используется слово «совпадающий». При применении к типам данных оно означает, что в настоящем стандарте определены операции равенства (в других контекстах применения данной системы кодирования слово «совпадающий» может иметь другие значения).

Примечание — Атрибут updateMode не воздействует на семантику или поведение самого типа данных, но может управлять поведением систем, обрабатывающих объекты, содержащие экземпляры типа данных.

7.3.3.3.3 flavorId : Set(String): сигнализирует о наложении на тип данных одного или нескольких множеств ограничений. Единственное назначение указания того, что существует дополнительное ограничение типа данных, заключается в обеспечении проверки экземпляра: средства контроля могут найти правила, заданные с помощью атрибута flavorId, и подтвердить, что экземпляр удовлетворяет этим правилам. Никакая другая обработка не должна зависеть от значения атрибута flavorId.

Никакая другая семантика или вычислительное применение не должны зависеть от значения этого свойства. Если оно не пусто, то должно быть допустимым ограничением типа данных значения.

Дополнительное обсуждение использования атрибутов тонкостей типов данных и атрибута flavorId приведены в разделе А.3.

7.3.3.4 Равенство equals

7.3.3.4.1 По умолчанию равенство equals определяется в соответствии с нижеизложенным. Отдельные специализации типа данных ANY переопределяют равенство equals в целях указания семантики вычисления равенства для этих специализаций. Для каждого типа данных точно документируется вычисление равенства equals.

7.3.3.4.2 В таблице 7 суммируются отношения между значениями null, nullFlavor и равенством equals.

Таблица 7 — Отношения между значениями null, nullFlavor и равенством equals

Другое значение / Это значение	null	nullFlavor	Правильное значение
null	null	null	null
nullFlavor	null	nullFlavor[1)]	nullFlavor[1)]
Правильное значение	null	nullFlavor[2)]	правильное значение[2)]

[1)] Первая общая генерализация обоих причин пустоты nullFlavor.
[2)] Если особо не оговорено для специализации, используется общий алгоритм равенства.

Согласно общему алгоритму равенства equals два значения равны, если они имеют одинаковый типа данных и если равны все атрибуты, не унаследованные от типов данных HXIT и ANY. Если какой-либо из приравниваемых атрибутов имеет пустое значение null или причину пустоты nullFlavor, то результат равен пустому значению null или первой общей генерализации причин пустоты (см. комментарии к сравнению причин пустоты в 7.1.4.

Операция равенства equals является рефлексивной, симметричной, транзитивной и постоянной, и все реализации должны удовлетворять этим требованиям. Правила равенства equals, определенные в настоящем стандарте, этим требованиям удовлетворяют.

Операция equals удовлетворяет общим правилам, описанным для операций и для причин пустоты, приведенным в 7.1.4, но здесь для большей ясности эти правила описаны более глубоко. В частности, правила транзитивны при применении операции равенства equals: если какой-либо из атрибутов или элементов коллекции имеет причину пустоты nullFlavor (отличающуюся от причины NA), то результатом будет то же самое значение причины пустоты nullFlavor, если иное особо не оговорено.

Операции равенства equals не переопределяют ни операцию «=», определенную в языке OCL, ни обычный эквивалент последней, используемой в любой платформе реализации. Она определяет семантическое равенство.

Примечание — При проверке равенства атрибуты updateMode и flavorId всегда игнорируются.

7.3.3.5 Инварианты

Экземпляр может иметь тип данных ANY (а не его специализацию), если у него есть причина пустоты nullFlavor, у которой не подразумевается значение INV.

Представление инвариантов на языке OCL:

```
  def: let isNull : Boolean = nullFlavor.oclIsDefined
  def: let isNotNull : Boolean = not isNull
  def: let noUpdate : Boolean = updateMode.oclIsUndefined
  def: let noUpdateOrHistory() : Boolean = noUpdate and
     noHistory
  def: let noUpdateOrHistory(member : ANY) : Boolean =
     member.oclIsDefined implies member.noUpdate and member.noHistory
  def: let invalid : Boolean = (nullFlavor = NullFlavor.INV or
       nullFlavor = NullFlavor.OTH or nullFlavor = NullFlavor.PINF or
       nullFlavor = NullFlavor.NINF or nullFlavor = NullFlavor.UNC or
       nullFlavor = NullFlavor.DER)

  inv "тип данных ANY разрешен только при наличии причины пустоты nullFlavor":
(isNotNull or invalid) implies
         not oclIsTypeOf(ANY)
```

7.3.3.6 Операции

7.3.3.6.1 equals(other : ANY) : BL: указывает, считается ли это значение данных семантически равным другому значению данных (other), то есть имеют ли одинаковый смысл. Обсуждение определения этой операции см. в 7.3.3.4.

7.3.3.6.2 isNull() : Boolean: указывает, имеет ли это значение причину пустоты nullFlavor или нет.

Эта операция является исключением из общих правил, описанных для операций и для причин пустоты: она возвращает true, если значение имеет причину пустоты nullFlavor, и false в противном случае.

7.3.3.6.3 notNull() : Boolean: указывает, имеет ли это значение причину пустоты nullFlavor или нет.

Эта операция является исключением из общих правил, описанных для операций и для причин пустоты: она возвращает true, если значение не имеет причину пустоты nullFlavor, и false в противном случае.

7.3.3.7 Примеры

7.3.3.7.1 Неизвестное значение

```
<example xsi:type="ANY" nullFlavor="UNK"/>
```

Значение не известно, и не известен также тип значения (он не указан в экземпляре и не определяется контекстом).

7.3.3.7.2 Пустое значение null и причина пустоты nullFlavor

Ниже приведено представление кодированного понятия как значения простого типа данных (см. 7.5.1).

```
<example code="784.0" codeSystem="2.16.840.1.113883.6.42">
  <displayName value="Headache"/>
</example>
```

У этого значения заданы атрибуты code, codeSystem и displayName. Все другие атрибуты пусты. В смысловых терминах это значение идентично следующему:

```
<example code="784.0" codeSystem="2.16.840.1.113883.6.42">
  <displayName value="Headache"/>
  <translation nullFlavor="NI"/>
</example>
```

Они имеют один и тот же смысл, поскольку в случае пустоты атрибута о нем можно сказать только то, что о его значении нет информации и причина этого неизвестна, то есть утверждать ровно то, что предусмотрено в значении пустоты nullFlavor.NI: нет информации, почему это значение не является правильным.

Аналогичным образом тот же смысл имеет следующее значение:

```
< example code="784.0" codeSystem="2.16.840.1.113883.6.42">
  <displayName value="Headache"/>
  <translation nullFlavor="NI" codeSystem="2.16.840.1.113883.6.96"/>
</example>
```

В данном случае есть определенное отличие: преобразование translation существует. Однако нет информации ни о нем, ни о том, почему его значение здесь отсутствует. Поэтому в смысловых терминах результат тот же.

В следующем примере показано, что для атрибута с причиной пустоты nullFlavor вполне можно предоставить дополнительную информацию:

```
< example code="784.0" codeSystem="2.16.840.1.113883.6.42">
  <displayName value="Headache"/>
  <translation nullFlavor="NI" codeSystem="2.16.840.1.113883.6.96"/>
</example>
```

В этом выражении указано несколько иное, а именно, что нет информации о преобразовании данного кода в систему кодирования SNOMED-CT. В отдельных случаях такая информация может оказаться значимой, хотя такое бывает редко.

7.3.3.7.3 Атрибут updateMode

Атрибут режима изменений updateMode принципиально важен при взаимодействии тесно связанных систем передачи и обработки сообщений. Тесно связанной системой является та, у которой ограниченное число приложений используется по строго определенным соглашениям между партнерами, а поток информации, передаваемой и принимаемой этими приложениями, хорошо организован и понятен. В этих условиях может оказаться выгоднее договориться, что вместо передачи в каждом сообщении всей имеющейся информации передаются только сведения об изменениях, произошедших в каждой транзакции. Это дает то преимущество, что стоимость реализации обмена информацией существенно снижается как для отправителя, так и для получателя, но при этом возрастает риск потери искажения информации, если порядок получения сообщений нарушен. Для правильного указания той информации, которая изменилась в транзакции, требуется атрибут updateMode.

В более общем случае, например для передачи медицинских документов или электронных медицинских карт, подобная обработка транзакций не применима, поэтому в таких контекстах атрибут updateMode обычно должен иметь фиксированное значение null.

Ниже показано несколько примеров того, как атрибут updateMode может использоваться в тесно связанных системах передачи и обработки сообщений. Эти примеры не должны рассматриваться как рекомендуемые руководства по использованию атрибута updateMode и соответствующей обработке, основанной на транзакциях.

В первом наборе примера рассмотрен простой случай, когда объектная модель имеет простой атрибут даты рождения лица birthDate:TS.

Пользователь открывает диалоговую форму для редактирования данных пациента и изменяет дату рождения на 21 июня 1975 года. Так как в системе уже хранится дата рождения, то она заменяется. Это приводит к тому, что тесно связанному целевому приложению передается экземпляр, содержащий следующий фрагмент:

```
<birthDate value="19750621" updateMode="R"/>
```

Приложение проверяет полученную информацию, определяет, что у него дата рождения уже хранится и заменяет ее на вновь полученное значение. Если бы прежней даты рождения не было, то это означало бы ошибку, хотя на практике обработка такой ситуации зависит от местных соглашений по безопасной обработке подобных случаев.

Позже другой пользователь открывает диалоговую форму для редактирования данных пациента и удаляет дату рождения. Это приводит к тому, что тесно связанному целевому приложению передается экземпляр, содержащий следующий фрагмент:

```
<birthDate nullFlavor="NI" updateMode="D"/>
```

Следует обратить внимание, что для придания этому значению допустимости требуется причина пустоты nullFlavor. Данный фрагмент означает, что дата рождения должна быть удалена и заменена на пустое значение с причиной пустоты NI. Это эквивалентно фрагменту:

```
<birthDate nullFlavor="NI" updateMode="R"/>
```

означающему, что дата рождения должна быть заменена на пустое значение с причиной пустоты NI. Как видно из этого примера, в подобных случаях в атрибуте updateMode нет особой необходимости. Он становится полезным, когда необходимо различать информацию, переданную в связи с ее изменением (UpdateMode.R), от информации, переданной для обеспечения идентификации. Для иллюстрации этого предположим, что в соглашении о транзакционной обработке было указано, что при изменении записи о пациенте в качестве идентификационных данных всегда должны передаваться фамилия, имя, отчество, пол и дата рождения. В этом случае экземпляр всегда содержит дату рождения. Используя значение атрибута updateMode, равное «К» (ключ), можно отметить, что атрибут даты рождения birthDate передан для идентификации, а не в связи с его изменением:

```
<birthDate nullFlavor="NI" updateMode="K"/>
```

Атрибут updateMode начинает приносить реальную пользу при управлении списками. В следующей группе примеров это демонстрируется на списке контактной информации пациента.

В типичном пользовательском интерфейсе для детализации разных видов контактной информации используются разные группы полей ввода. В информационную эру конкретные виды контактной информации, которые необходимо обеспечивать в интерфейсе, имеют тенденцию меняться, поэтому в настоящем стандарте детализация контактной информации представлена в виде типизированного списка (все больше и больше приложений следуют этому подходу). В данном гипотетическом случае используются отдельные поля ввода для номера стационарного телефона, номера мобильного телефона, номера факса и адреса электронной почты, а результатом ввода является атрибут contacts : DSET(TEL). Типичная запись пациента может содержать список контактной информации наподобие следующего, передаваемого в нетранзакционной среде:

```
<contacts>
  <item value="tel:+11015551234" use="H" capabilities="voice"/>
  <item value="tel:+11995556787" use="MC" capabilities="voice sms"/>
  <item value="tel:+11015551235" capabilities="fax"/>
  <item value="                                    >
</contacts>
```

gost.e-ecolog.ru

В тесно связанной системе, использующей транзакционную обработку на основе атрибута updateMode, такой полный список редко будет передаваться. Вместо этого чаще будет передаваться только его часть.

Если пользователь открыл экранную форму с данными о пациенте и изменил в ней номер домашнего телефона, то может быть передан следующий экземпляр:

```
<contacts>
  <item value="tel:+11015551234" use="H"
      capabilities="voice" updateMode="D"/>
  <item value="tel:+12315559876" use="H"
      capabilities="voice" updateMode="A"/>
</contacts>
```

Приложение-получатель сравнивает имеющуюся у него информацию с этим экземпляром. Если оно не может найти исходный номер +11015551234, то это ошибка, которая должна быть обработана в соответствии с местными соглашениями. Если оно нашло этот номер, то удаляет его, а затем добавляет новый номер. Вместо указанного выше экземпляра можно попытаться послать другой, а именно:

```
<contacts>
  <item value="tel:+12315559876" use="H"
      capabilities="voice" updateMode="R"/>
</contacts>
```

с инструкцией о замене существующего номера. Однако какой номер следует заменить? На это нет ответа, поскольку совпадение основано на равенстве, значит, этот фрагмент означает замену номера домашнего телефона +12315559876 на номер +12315559876.

Поэтому лучше использовать операции удаления (*D*) и добавления (*A*), указывая значения типа данных. Операции замены (*R*) и добавления/замены (*AR*) не очень полезны: поскольку совпадение основано на равенстве, все, что можно сказать, это заменить значение *A* на значение *A*, что не имеет особого смысла. В общем случае полезнее использовать сложные объекты, включаемые в список допустимых значений типа данных, тем самым обеспечивая удовлетворение всех известных вариантов транзакционной обработки.

Существует пример, в котором можно использовать значение UpdateModel.R. Поскольку совпадение основано на равенстве, в котором используется передаваемый номер, то следующие два экземпляра несут один и тот же смысл:

```
<contacts>
  <item value="tel:+1.01555:234" use="H"
      capabilities="voice" upateMode="D"/>
  <item value="tel:+ 110.5551234" use="H WP"
      capabilities="voice fax" updateMode="A"/>
</contacts>

<contacts>
  <item value="tel:+ 110:5551234" use="H W"
      capabilities="voice fax" updateMode="R"/>
</contacts>
```

Оба этих экземпляра означают, что существующие детали использования номера +12315559876, а именно use (назначение) и capabilities (функции), надо заменить на новые.

7.3.4 Тип данных BL

7.3.4.1 Описание

Специализация типа данных ANY

Тип данных BL используется для значений двузначной логики. Значение типа BL может быть равно true или false либо может иметь причину пустоты nullFlavor.

7.3.4.2 Синтаксис ИСО/МЭК 11404

```
type BL = class (
  validTimeLow : characterstring,
  validTimeHigh : characterstring,
  controlInformationRoot : characterstring,
  controlInformationExtension : characterstring,
  nullFlavor : NullFlavor,
  updateMode : UpdateMode,
  flavorId : Set(characterstring),
  value : boolean
)
```

Поскольку любое значение данных может иметь причину пустоты nullFlavor, двузначная логика в действительности расширяется до трехзначной в соответствии с таблицей 8.

Таблица 8 — Таблица истинности

NOT			AND	true	false	null		OR	true	false	null
true	false		true	true	false	null		true	true	true	true
false	true		false	false	false	false		false	true	false	null
null	null		null	null	false	null		null	true	null	null

В этой таблице null означает либо настоящее пустое значение null, либо пустое значение с причиной пустоты nullFlavor. Если это настоящее пустое значение (или одно из нескольких настоящих пустых значений), то результатом будет настоящее пустое значение. Если булевская операция выполняется над двумя значениями типа данных BL с разными причинами пустоты nullFlavor, то значением причины пустоты результата должно быть значение какого-либо общего родителя двух различных причин пустоты nullFlavor. Рекомендуется в качестве результата использовать первого общего родителя.

7.3.4.3 Атрибуты

7.3.4.3.1 value : Boolean: значение типа BL

Это пример шаблона обертки примитивного типа. Дополнительная информация приведена в 6.3.

7.3.4.4 Равенство

Два булевских значения равны, если они не пусты (свойство nonNull равно true) и имеют одно и то же значение.

7.3.4.5 Инварианты

Если причина пустоты nullFlavor не указана, то экземпляр типа данных BL должен иметь значение.

Представление инвариантов на языке OCL:

```
inv "value if not nullFlavored":
            isNotNull implies value.oclIsDefined
```

7.3.4.6 Операции

7.3.4.6.1 and(other : BL) : BL: результат равен true, если оба значения равны true Результат равен false, если хотя бы одно из значений равно false. В остальных случаях результат равен null или имеет причину пустоты nullFlavor.

7.3.4.6.2 or(other : BL) : BL: результат равен true, если данное значение или значение параметра other равно true Результат равен false, если оба значения равны false. В остальных случаях результат равен null или имеет причину пустоты nullFlavor.

7.3.4.6.3 xor(other : BL) : BL: результат равен true, если данное значение и значение параметра other различны. Результат равен false, если оба эти значения одинаковы. В остальных случаях результат равен null или имеет причину пустоты nullFlavor.

7.3.4.6.4 implies(other : BL) : BL: результат равен true, если данное значение равно false или данное значение и значение параметра other равны true. Результат равен false, если данное значение равно true, и значение параметра other равно false. В остальных случаях результат равен null или имеет причину пустоты nullFlavor.

7.3.4.6.5 not() : BL: результат равен false, если данное значение равно true. Результат равен true, если данное значение равно false. В остальных случаях результат равен null или имеет причину пустоты nullFlavor.

Примечание — В силу особой природы этих логических операций они не всегда следуют общим правилам, описанным для значений причин пустоты nullFlavor и операций. Например, выражение «(null/nullFlavored) or true» всегда имеет значение true, поскольку безразлично, равно ли отсутствующее значение true или false: результат будет одинаков. Кроме того, с точки зрения реализации эти операции строго симметричны, и для предотвращения ошибок реализации при выполнении операций над пустыми объектами null может потребоваться некоторая специальная обработка.

7.3.4.7 Примеры

7.3.4.7.1 Простое значение true

```
<example xsi:type="BL" value="true"/>
```

7.3.4.7.2 Неизвестное значение

```
<example xsi:type="BL" nullFlavor="UNK"/>
```

7.3.5 BL.NONNULL

7.3.5.1 Описание

Атрибут тонкости, ограничивающий тип данных BL

BL.NONNULL представляет собой ограниченный экземпляр типа BL, у которого не может быть причины пустоты nullFlavor. Отсюда вытекает, что вместо BL.NONNULL никогда не может быть использовано значение null, хотя это и не является правилом, которое явно требуется настоящим стандартом.

7.3.5.2 Инварианты

BL.NONNULL не может иметь причину пустоты nullFlavor.

Представление инвариантов на языке OCL:

```
inv "cannot have a nullFlavor": not isNull
```

7.4 Текстовые и двоичные типы данных

7.4.1 Введение

Типы данных, предназначенные для текстовых и мультимедийных данных.

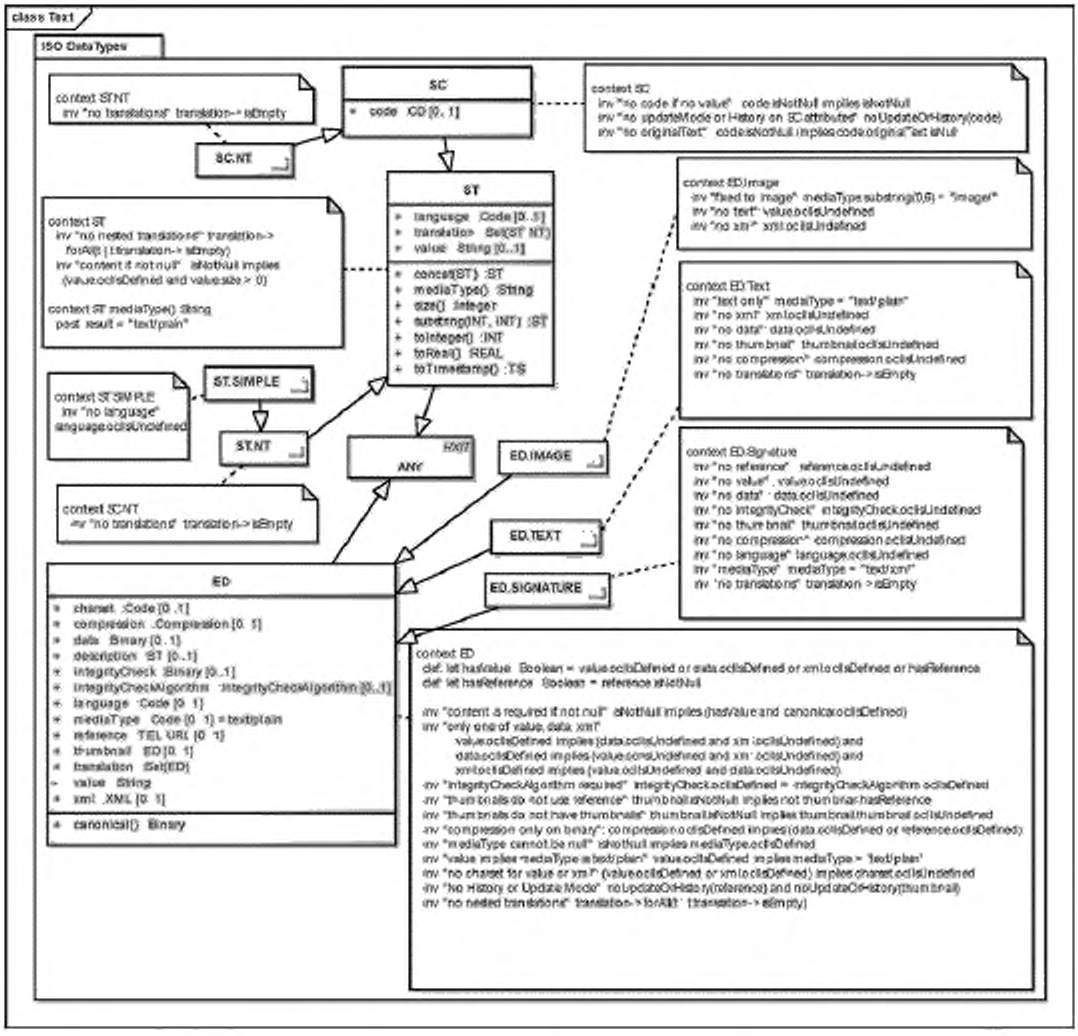


Рисунок 3 — Текстовые и двоичные типы данных

7.4.2 Тип данных ED (Encapsulated Data — инкапсулированные данные)

7.4.2.1 Описание

Специализация типа данных ANY

Данные, в первую очередь предназначенные для интерпретации человеком или дальнейшей машинной обработки, не входящей в область применения настоящего стандарта. К ним относятся форматированный или неформатированный текст на человеческом языке, мультимедийные данные или структурированная информация, определенная в другом стандарте (например, подписи XML).

Инкапсулированные данные могут присутствовать в двух формах: по значению или по ссылке. Их содержание является тем же самым вне зависимости от того, присутствуют ли они как местные, по значению, или как удаленные, по ссылке. Местные данные передаются или перемещаются как часть значения инкапсулированных данных, в то время как ссылочные данные могут находиться в другом месте: в качестве ссылки на эти данные используется идентификатор URL или URI. Данные, передаваемые по значению, могут быть представлены тремя разными способами:

- как последовательность символов (в атрибуте value);

- как двоичные данные (последовательность байтов) (в атрибуте data);
- как xml-содержание (в атрибуте xml).

Если у значения типа данных ED не задан атрибут причины пустоты nullFlavor, то оно должно иметь непустое содержание. Оно может быть предоставлено по значению (в одном из атрибутов value, data или xml) или по ссылке. Содержание может быть представлено одновременно и по ссылке, и по значению, в этом случае оба экземпляра содержания должны быть идентичными. Элементы обработки информации не обязаны проверять это условие, но могут рассматривать несовпадение как ошибку.

7.4.2.2 Синтаксис ИСО/МЭК 11404

```
type ED = class (
  validTimeLow : characterstring,
  validTimeHigh : characterstring,
  controlInformationRoot : characterstring,
  controlInformationExtension : characterstring,
  nullFlavor : NullFlavor,
  updateMode : UpdateMode,
  flavorId : Set(characterstring),
  value : characterstring,
  data : Sequence(Octet),
  xml : XML,
  reference : TEL.URL,
  mediaType : characterstring,
  charset : characterstring,
  language : characterstring,
  compression : Compression,
  integrityCheck : Sequence(Octet),
  integrityCheckAlgorithm : IntegrityCheckAlgorithm,
  description : ST,
  thumbnail : ED,
  translation : Set(ED)
)
```

7.4.2.3 Атрибуты

7.4.2.3.1 value : String: простая последовательность символов, представляющая содержание.

Если используется атрибут value, то атрибут mediatype имеет фиксированное значение text/plain и набор символов должен быть совместим с набором символов типа данных String. Дополнительная информация представлена в 6.7.5.

7.4.2.3.2 data : Binary: простая последовательность байтов, представляющая содержание.

7.4.2.3.3 xml : XML: содержание в формате XML.

Данный атрибут имеет непосредственное представление на языке XML. Это обусловлено тем, что настоящий стандарт предусматривает сериализацию данных на языке XML, и этот атрибут xml специально предназначен для обработки в сериализованной форме. Семантически значение атрибута xml ничем не отличается от представления тех же данных в атрибуте value или data.

Примечание — Эти три представления данных типа ED — как последовательность символов, последовательность байтов или на языке XML в нативном формате XML format — взаимно несовместимы и должны были бы быть реализованы как три специализации абстрактного родительского типа ED. Однако это значительно усложнило бы определение и реализацию тонкостей типа данных ED, а также соответствующий XML-формат (его потребовалось бы дополнить обязательным атрибутом xsi:type) и при этом не упростило бы заметно общую реализацию типа данных ED.

7.4.2.3.4 reference: TEL.URL : адрес URL, по которому находится двоичное содержание.

Семантическое значение инкапсулированных данных остается тем же самым вне зависимости от того, присутствует ли содержание по значению или только по ссылке. Однако ссылочные инкапсулированные данные ведут себя иначе, поскольку при любой попытке проверить содержание требуется загрузить их с этого адреса.

Инкапсулированные данные могут одновременно присутствовать и по значению, и по ссылке.

Если данные присутствуют в атрибуте value, data или xml, то ссылка должна указывать на те же самые данные. Ситуация, когда данные, полученные по ссылке, не проходят проверку целостности, или не совпадают с данными, присутствующими по значению, или не совпадают с данными, которые были получены по этой ссылке раньше и сохранились в кэше, является ошибкой. Значение атрибута mediatype типа данных ED должно совпадать с типом среды, возвращаемом при доступе к ссылке.

Ссылка может содержать атрибут usablePeriod, указывающий, что данные могут быть доступны только ограниченный период времени. Вне зависимости от того, ограничен доступ к ссылке периодом usablePeriod или нет, содержание ссылке должно оставаться одним и тем же все время. Любое приложение, использующее эти данные по ссылке, всегда должно получать те же самые данные или ошибку. Ссылка не может быть повторно использована для получения другой версии этих данных или других данных.

7.4.2.3.5 mediaType : Code: идентифицирует тип среды инкапсулированных данных и может использоваться для определения метода интерпретации или изображения содержания этих данных.

Домен типов среды, определенных организацией IANA, описан в документах RFC 2045 и 2046 организации IETF. По умолчанию атрибут mediaType имеет значение text/plain и не может быть пустым. Если тип среды отличается от text/plain, то атрибут mediaType должен иметь значение.

Если содержание сжато с помощью заданного алгоритма сжатия, то значение атрибута mediaType должно указывать тип среды данных, имевшийся до сжатия, вне зависимости от того, присутствуют данные по ссылке или нет.

7.4.2.3.6 charset : Code: код, присвоенный набору символов и кодировке символьной информации организацией Internet Assigned Numbers Authority (IANA) и положениями документа RFC 2978 [http://

gost.e-ecolog.ru

].

Свойство charset должно быть известно, если значение типа ED представляет собой символьные данные в любой форме. Если данные присутствуют непосредственно в атрибуте value, то набор символов должен быть известным и совместимым с набором символов типа данных String (дополнительную информацию см. в 6.7.5). Если эти данные предоставляются по ссылке и метод доступа не обеспечивает идентификацию набора символов содержания (обычно возвращаемую в заголовке MIME), то эта идентификация должна быть указана как компонент типа данных *ED*.

7.4.2.3.7 language : Code: человеческий язык содержания. Допустимые коды берутся из документа RFC 3066 организации IETF. Если этот атрибут пуст, то язык может быть идентифицирован откуда-то еще, например, из содержания или из тега языка в кодировке unicode.

В профилях соответствия должны быть определены правила задания языка по умолчанию, применяемые в данной среде использования настоящего стандарта.

Примечание — Хотя значение атрибута language обычно изменяет интерпретацию текста, содержащийся в нем код языка не меняет значение символов текста.

7.4.2.3.8 compression : Compression: алгоритм сжатия, если таковой был применен к исходным байтовым данным. Если этот атрибут пуст, то данные не являются сжатыми. Сжатие применяется только к двоичной форме содержания. Если он имеет значение, то оно должно быть взято из системы кодирования HL7 CompressionAlgorithm (см. таблицу 9).

Таблица 9 — Перечисление CompressionAlgorithm, ОИД: 2.16.840.1.113883.5.1009

Уровень	Код	Описание	Определение
1	DF	deflate	Формат дефляционного сжатия, предложенный в документе RFC 1951
1	GZ	gzip	Формат сжатия данных, совместимый с широко используемой утилитой GZIP и описанный в документе RFC 1952 (использует алгоритм дефляции)
1	ZL	zlib	Сжатый формат данных, также использующий алгоритм дефляции. Описан в документе RFC 1950
1	Z	compress	Оригинальный алгоритм сжатия и формат файла, предложенный в операционной системе UNIX и использующий алгоритм LZC (вариант алгоритма LZW). Отягощен патентами и менее эффективен по сравнению с алгоритмом дефляции

Окончание таблицы 9

Уровень	Код	Описание	Определение
1	BZ	bzip	Формат сжатия bzip-2. Дополнительная информация приведена на сайте http://gost.e-ecolog.ru
1	Z7	Z7	Формат сжатия файла 7z. Дополнительная информация приведена на сайте http://www.7-zip.org/7z.htm

Некоторые форматы сжатия позволяют включать в один архивный сжатый файл несколько исходных файлов. Приложения должны проверять, что исходные данные, полученные после разархивирования, соответствуют заявленному типу среды.

Синтаксис ИСО/МЭК 11404 атрибута сжатия:

```
type Compression = enumeration (DF, GZ, ZL, Z, BZ, Z7)
```

7.4.2.3.9 integrityCheck : Binary: контрольная сумма двоичных данных. Это свойство предназначено для использования при передаче ссылок на данные и позволяет позже проверить, не изменились ли ссылочные данные после того, как было создано инкапсулированное значение, ссылающееся на эти данные. Если этот атрибут пуст, то такая проверка целостности не выполняется.

Если данные, извлеченные по ссылке, не проходят проверку целостности, это должно признаваться ошибкой.

Контрольная сумма integrityCheck вычисляется с помощью алгоритма, указанного в атрибуте integrityCheckAlgorithm. По умолчанию должен использоваться алгоритм Secure Hash Algorithm-1 (SHA-1). Контрольная сумма представляется в двоичном виде в соответствии с правилами, принятыми в алгоритме контрольного суммирования.

Контрольная сумма вычисляется по двоичным данным, содержащимся в компоненте данных или доступным по ссылке. Перед вычислением контрольной суммы никаких преобразований не делается. Если данные сжаты, то контрольная сумма вычисляется по сжатым данным.

7.4.2.3.10 integrityCheckAlgorithm : IntegrityCheckAlgorithm: алгоритм, использованный для вычисления значения атрибута integrityCheck.

Если этому атрибуту присвоено значение, то оно должно быть взято из системы кодирования HL7 IntegrityCheckAlgorithm. Текущие допустимые значения указаны в таблице 10.

Таблица 10 — Перечисление IntegrityCheckAlgorithm, ОИД 2.16.840.1.113883.5.1010

Код	Описание	Определение
SHA-1	secure hash algorithm-1	Этот алгоритм определен в документе FIPS PUB 180-1: Secure Hash Standard (по состоянию на 17 апреля 1995 года)
SHA-256	secure hash algorithm-256	Этот алгоритм определен в документе FIPS PUB 180-2: Secure Hash Standard

Синтаксис ИСО/МЭК 11404 атрибута integrityCheckAlgorithm:

```
type IntegrityCheckAlgorithm = enumeration (SHA1, SHA256)
```

7.4.2.3.11 description : ST: альтернативное описание мультимедийных данных, используемое, если контекст не позволяет отобразить эти данные.

Например, краткое текстовое описание изображения или аудиоклипа и т. д. Этот атрибут не предназначен для полной подстановки вместо оригинала. Для полной подстановки используйте свойство «translation» (перевод).

Этот атрибут предназначен для удовлетворения требований передачи информации о нетрудоспособности, например, тех, которые описаны в законе Americans with Disability Act (известном также как «Section 508»), требующем, чтобы вложенные мультимедийные данные сопровождались кратким

текстовым описанием, которое может быть прочтено в экранной форме. Это похоже на очень краткий «эскиз» для типа среды mediaType = text/plain.

7.4.2.3.12 thumbnail : ED: сокращенная форма полного содержания. Этот атрибут содержит сокращенное представление полных данных (эскиз), требующее гораздо меньше ресурсов, нежели полные данные, но при этом обладающее отличительными способностями полных данных. Обычно эскизы используются для ссылочных инкапсулированных данных. Они позволяют пользователю выбирать нужные данные, не загружая по ссылке их полное представление.

Первоначально термин «эскиз» (thumbnail) использовался для обозначения изображения в более низком разрешении (или меньшего размера) по сравнению с другим изображением. Однако это понятие может быть метафорически использовано для других типов данных, кроме изображений. Например, видеозапись может быть представлена коротким клипом, аудиоклип другим аудиоклипом, более коротким, имеющим меньшую частоту оцифровки или использующим искажающее сжатие.

Эскиз не может содержать вложенный эскиз.

7.4.2.3.13 translation : Set(ED): перевод, то есть альтернативное представление того же самого содержания на другом языке или в другом типе среды mediaType.

Переводы должны представлять ту же самую информацию, но на другом языке или в другом типе среды mediaType. Переводы не могут содержать другие переводы. Переводы не принимают участие в проверке на равенство и, соответственно, не должны добавлять какую-либо новую семантику к исходному значению.

7.4.2.4 Равенство

Два значения типа ED равны только том случае, если у них одинаковы типы среды mediaType и данные. Если значения типа ED содержат ссылочные или сжатые (упакованные) данные, то в проверке на равенство должны участвовать данные, извлеченные по ссылке и распакованные (каноническое содержание, см. 7.4.2.6.1). Свойства сжатия compression, эскиза thumbnail и ссылки reference в проверке на равенство не участвуют. Кроме того, из этой проверки исключается свойство языка language (см. 7.4.2.7.2). Если типом среды mediaType являются символьные данные и при этом свойства набора символов (charset) не одинаковы, то перед проверкой надо отобразить данные на общий набор символов.

Значение типа данных ED, у которого содержание имеет тип среды mediaType, равный text/plain, может быть также равно значению типа данных ST при условии выполнения описанных выше правил.

7.4.2.5 Инварианты:

- один из атрибутов reference, data, value или xml должен присутствовать, и хотя бы один байт данных должен быть доступен по ссылке, если значение типа ED не имеет причины пустоты nullFlavor;
- может присутствовать только один из атрибутов data, value или xml.
- атрибут integrityCheckAlgorithm должен быть указан только в том случае, если присутствует атрибут integrityCheck;
- если присутствует атрибут эскиза thumbnail, то он не должен включать в себя атрибут ссылки reference;
- если присутствует атрибут эскиза thumbnail, то он не должен включать в себя атрибут эскиза thumbnail;
- атрибут сжатых данных compression может быть указан только в том случае, если данные являются двоичными или предоставляются по ссылке;
- атрибут типа среды mediatype не может быть пустым,
- если значение передается в атрибуте value, то атрибут типа среды имеет значение plain/text;
- набор символов не должен объявляться для содержания, передаваемого в виде текста или в формате XML (описание текстового формата см. в 6.7.5, а в формате XML предусмотрено внутреннее указание набор символов).

Представление инвариантов на языке OCL:

```
def: let hasValue : Boolean = value.oclIsDefined or
    data.oclIsDefined or xml.oclIsDefined or hasReference
def: let hasReference : Boolean = reference.isNotNull
inv "если содержание не пусто, то оно требуется": isNotNull implies (hasValue
    and canonical.oclIsDefined)
inv "только один из атрибутов value, data, xml": value.oclIsDefined implies
```

```
    (data.oclIsUndefined and xml.oclIsUndefined) and data.oclIsDefined implies
      (value.oclIsUndefined and xml.oclIsUndefined) and xml.oclIsDefined
            implies
                  (value.oclIsUndefined and data.oclIsUndefined)
inv "алгоритм integrityCheckAlgorithm требуется": integrityCheck.oclIsDefined
    = integrityCheckAlgorithm.oclIsDefined
inv "эскизы не содержат ссылки": thumbnail.isNotNull implies
    not thumbnail.hasReference
inv "эскизы не содержат эскизы": thumbnail.isNotNull implies
    thumbnail.thumbnail.oclIsUndefined
inv "сжатие применяется только к двоичным данным":
    compression.oclIsDefined implies
      (data.oclIsDefined or reference.oclIsDefined)
inv "тип среды mediaType не может быть пуст": isNotNull implies
    mediaType.oclIsDefined
inv "наличие атрибута value подразумевает тип среды mediaType, равный text/
plain": value.oclIsDefined implies
    mediaType = 'text/plain'
inv "для атрибутов value или xml не должен быть указан атрибут charset":
    (value.oclIsDefined or xml.oclIsDefined) implies
    charset.oclIsUndefined
inv "режим истории или изменений запрещен ": noUpdateOrHistory(reference)
    and noUpdateOrHistory(thumbnail)
inv "no nested translations": translation >forAll(t |
    t.translation >isEmpty)
```

7.4.2.6 Операции

7.4.2.6.1 canonical . Binary: последовательность байтов, представляющая фактические данные.

Примечания

1 Последовательность байтов извлекается по ссылке, если данные предоставлены по ссылке, и распакуется, если использовалось сжатие.

2 Эта операция не следует обычным правилам, применяемым к операциям и причине пустоты nullFlavor, поскольку тип возвращаемого значения не может иметь причину пустоты nullFlavor.

3 Если значение типа данных ED имеет причину пустоты nullFlavor, то операция возвращает пустое значение.

7.4.2.7 Примеры

7.4.2.7.1 Простой текст

```
<example xsi:type="ED" value="это простой текст" mediaType="text/plain"/>
<example xsi:type="ED" value="это простой текст"/>
```

Эти значения типа ED являются примерами простого текста. Типу среды mediaType присвоено значение text/plain. Поскольку оно является значением этого атрибута по умолчанию, то его не надо указывать, когда данные представляются на языке XML, и второй пример также правилен. Набор символов не указан; в данном случае набор символов не должен задаваться, он определяется заголовком XML.

7.4.2.7.2 Язык

```
<example xsi:type="ED" flavorId="ED.TEXT" value="this is plain text" lan
guage="en" mediaType="text/plain"/>
<example xsi:type="ED" flavorId="ED.TEXT" value="dieses ist normaler Text" lan
guage="de" mediaType="text/plain"/>
```

В первом случае язык явно задан как английский, во втором — как немецкий. Если атрибут language отсутствует, то язык не известен, хотя обычно безопасно считать, что используется наиболее широко распространенный местный язык. Типу присвоен атрибут тонкости ED.TEXT, то есть только текст.

```
<example xsi:type="ED" value="this is plain text" language="en">
    <translation xsi:type="ED" value="dieses ist normaler Text" language="de"/>
</example>
```

Здесь немецкая версия присутствует как перевод английской версии.

```
<example xsi:type="ED" value="this is plain text" language="en-ca">
    translation xsi:type="ED" value="ce ci est du texte non structure"
language="fr ca"/>
</example>
```

Может быть также указан локализованный язык, в данном примере — канадская версия французского языка.

7.4.2.7.3 Двоичное содержание

```
<example xsi:type="ED">
    <data>dGhpcyBpcyBiaW5hcnkgY29udGVudA==</data>
</example>
```

Это значение типа ED содержит простой текст «this is binary content» (это двоичное содержание). Поскольку атрибут типа среды mediaType имеет значение text/plain, то его указывать не надо, так как он имеет это значение по умолчанию.

Примечание — Набор символов в этом примере не известен; он не обязательно совпадает с тем, который использован в XML-значении, и предположение такого совпадения не является безопасным.

```
<example xsi:type="ED" charset="UTF 8">
    <data>dGhpcyBpcyBiaW5hcnkgY29udGVudA==</data>
</example>
```

По этой причине при использовании элемента данных data обычно стоит указывать набор символов, как это показано выше в примере.

7.4.2.7.4 Ссылка

```
<example xsi:type="ED" mediaType="image/jpg">
    <reference value="gost.e-ecolog.ru                    ">
</example>
```

Содержание этого значения типа ED находится по адресу “http://gost.e-ecolog.ru ”. При загрузке данных по этой ссылке возвращается двоичный поток байтов, в заголовке ответа HTTP будет указан тип среды «image/jpg».

Примечание — Протокол http непосредственно поддерживает сжатие данных. Однако атрибут compression типа данных ED имеет отношение не к тому сжатию, информация о котором возвращается в заголовке ответа HTTP, а к тем данным, которые получатся после того, как ответ HTTP будет завершен и интерпретирован.

```
<example xsi:type="ED" mediaType="image/jpg" compression="GZ">
    <reference value="gost.e-ecolog.ru                    ">
</example>
```

В этом примере указано, что результат ответа HTTP представляет собой версию байтов изображения, сжатых с помощью алгоритма gzip. В заголовке ответа HTTP также могло быть указано, что ответ HTTP сжат с помощью алгоритма gzip — это означало бы второе (избыточное) сжатие данные (хотя даже первое сжатие было бы избыточным, поскольку базовый тип данных — JPEG — обеспечивает весьма эффективное представление данных изображения).

7.4.2.7.5 Содержание на языке XML

```
<example xsi:type="ED" mediaType="text/xml">
```

```
    <data>PHBhcmVudD4NCiAgPGNoaWxkPlRoaXMgaXMgc29tZSB0ZXh0IGluIHRoZSBjaGlsZDwvY
2hpbGQ+DQogIFRoaXMgaXMgc29tZSB0ZXh0IGluIHRoZSBwYXJlbnQNCjwvcGFyZW50Pg==</data>
</example>
```

Это значение типа ED представляет собой некоторое содержание на языке XML, представленное в двоичной форме. Как и в предыдущем примере, набор символов XML-содержания не известен; оно не обязательно совпадает с тем, и которое используется в данном значении типа ED, и предположение о таком совпадении не является безопасным.

```
<example xsi:type="ED" mediaType="text/xml" charset="ASCII">
    <data>PHBhcmVudD4NCiAgPGNoaWxkPlRoaXMgaXMgc29tZSB0ZXh0IGluIHRoZSBjaGlsZDwvY
2hpbGQ+DQogIFRoaXMgaXMgc29tZSB0ZXh0IGluIHRoZSBwYXJlbnQNCjwvcGFyZW50Pg==</data>
</example>
```

В данном примере набор символов задан явно. Он не обязательно совпадает с кодировкой, указанной в XML-документе.

```
<example xsi:type="ED" mediaType="text/xml">
    <xml>
      <parent>
            <child>This is some text in the child</child>
      This is some text in the parent
     </parent>
    </xml>
</example>
```

В этом примере xml-содержание представлено в виде значения элемента xml. Значение атрибута типа среды mediaType этого содержания должно быть представлено (оно не может быть равно text/plain). Набор символов не может быть задан, поскольку он должен совпадать с тем, который указан в заголовке xml.

7.4.2.7.6 Другие типы содержания

```
<example mediaType="application/pdf" compression="GZ">
  <data>/9j/4AAQSkZJRgABAgEAgACAAAD/2wCEAAICAgIC
  AQICAgICAgICAwQDAwMDAwUEBAMEBgYHBgYG
  BgYHCAoIBwcJBwYGCAsJCQoKCwsLBwgMDQwKDAoLCwoBAgIC
  AwMDBQMDBQoHBgcKCgoKCgoKCgoK
  CgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgo
  KCv/AABEIAg4CuQMBIQACEQED
  EQH / xAG i AAABBQEBAQEBAQAAAAAAAAAAAQI DBAUGBwg JCgs BAAM
  BAQEBAQEBAQEAAAAAAAABAgME
  BQYHCAkKCxAAAgEDAwIEAwUFBAQAAAF 9AQIDAAQRBRIhMUEGElF
  hByJxFDKBkaEIIOKxwRVSOfAk
  ...
  CigAooAKKACigAooAKKACigAooAKKACigAooAKKACigAooAKKAC
  gAooAKKACigAooAKKACigAoo
  AKKACigAooAKKACigAooAKKACigAooAKKACigAooAKKACigAooAK
  KAC i gAooAKKACi gAooAKKACi
  gAooAKKACigAooAKKACigAooAKKACigAooAKKACigAooAKKACigA
  ooAKKACigAooAKKAP//Z</data>
</example>
```

В этом примере показано содержание документа в формате Acrobat фирмы Adobe, которое было сжато с помощью алгоритма GZip.

```
<example mediaType="image/png">
    <reference value="http://example.org/xrays/128s8d9ej229se32s.png">
```

```
      <useablePeriod xsi:type="IVL_TS">
            <low value="200007200845"/>
            <high value="200008200845"/>
      </useablePeriod>
    </reference>
    <integrityCheck>EQH/xAGiAAABBQEBAQEBAQAAAAAAAAAAAQIDBAUGBwgJCgsB
AAMBAQEBAQEBAQEAAAAAAAABAgME</integrityCheck>
    <thumbnail mediaType="image/jpeg">
     <data>/9j/4AAQSk 2 JRgABAgEAgACAAAD/2wCEAAICAgIC
     AQICAgICAgICAwQDAwMDAwUEBAMEBgYHBgYG
     BgYHCAoIBwcJBwYGCAsJCQoKCwsLBwgMDQwKDAoLCwoBAgIC
     AwMDBQMDBQoHBgcKCgoKCgoKCgoK
     CgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgo
     KCv/AABEIAg4CuQMBIQACEQED
     EQH / xAG 1 AAABBQEBAQEBAQAAAAAAAAAAAQI DBAUGBwg JCgs BAAM
     BAQEBAQEBAQEAAAAAAAABAgME
     BQYHCAkKCxAAAgEDAwIEAwUFBAQAAAF 9AQIDAAQRBRIhMUEGE1F
     hByJxFDKBkaLIOKxwRVSOfAk
     CigAooAKKACigAooAKKACigAooAKKACigAooAKKACigAooAKKAC
     gAooAKKACigAooAKKACigAoo
     AKKACi gAooAKKAC 1 gAooAKKAC 1 gAooAKKAC 1 gAooAKKAC 1 gAooAK
     KACigAooAKKACigAooAKKACi
     gAooAKKACigAooAKKACigAooAKKACigAooAKKACigAooAKKACigA
     ooAKKACigAooAKKAP//Z </data>
    </thumbnail>
</example>
```

Этот пример содержит ссылку на изображение, хранящееся по определенному адресу URL и доступное в течение месяца. Для этого изображения предусмотрены проверка целостности, а также эскиз, вложенный в данное значение типа ED.

7.4.3 ED.IMAGE

7.4.3.1 Описание

Тонкость, ограничивающая тип данных ED

Тонкость ED.IMAGE ограничивает тип данных ED таким образом, что содержание этого типа должно быть изображением.

7.4.3.1.1 Инварианты:

- значение атрибута типа среды mediaType должно начинаться со строки "image/";
- содержание не может быть представлено как текст или XML-документ — оно должно быть двоичным и представляться по значению и/или по ссылке.

Представление инвариантов на языке OCL:

```
inv "fixed to image": mediaType.substring(0,6) = "image/"
inv "no text": value.oclIsUndefined
inv "no xml": xml.oclIsUndefined
```

7.4.4 ED.TEXT

7.4.4.1 Описание

Тонкость, ограничивающая тип данных ED

Тонкость ED.TEXT ограничивает тип данных ED таким образом, что содержание этого типа должно быть простым текстом.

Эта тонкость полезна, поскольку иногда бывает необходимо предоставлять по ссылке значение, содержание которого должно быть простой строкой. Кроме того, не разрешаются переводы.

7.4.4.2 Инварианты:

- значение атрибута типа среды mediaType должно быть равно text/plain;
- содержание не может быть представлено как XML-документ или двоичные данные — оно должно быть текстом и представляться по значению и/или по ссылке;
- эскиз, сжатие и переводы не разрешены.

Представление инвариантов на языке OCL:

```
inv "text only": mediaType = "text/plain"
inv "no xml": xml.oclIsUndefined
inv "no data": data.oclIsUndefined
inv "no thumbnail": thumbnail.oclIsUndefined
inv "no compression": compression.oclIsUndefined
inv "no translations": translation >isEmpty
```

7.4.5 ED.SIGNATURE

7.4.5.1 Описание

Тонкость, ограничивающая тип данных ED

Тонкость ED.SIGNATURE ограничивает тип данных ED таким образом, что содержание этого типа должно быть электронной подписью XML, соответствующей рекомендациям XML Signature Syntax and Processing Recommendation организации W3C.

Если эта тонкость реализуется в контексте, объявляющем косвенное соответствие настоящему стандарту, то реализация может отличаться от рекомендаций организации W3C, и объявление соответствия должно описывать отображение между реализацией электронной подписи и рекомендациями организации W3C.

7.4.5.2 Инварианты:

- атрибуты value, data, reference, integrity check, thumbnail, compression, language и translation не разрешены;
- атрибут типа среды mediaType должен иметь значение text/xml.

Представление инвариантов на языке OCL:

```
inv "no reference" : reference.oclIsUndefined
inv "no value" : value.oclIsUndefined
inv "no data" : data.oclIsUndefined
inv "no integrityCheck": integrityCheck.oclIsUndefined
inv "no thumbnail": thumbnail.oclIsUndefined
inv "no compression": compression.oclIsUndefined
inv "no language": language.oclIsUndefined
inv "mediaType": mediaType = "text/xml"
inv "no translations": translation->isEmpty
```

7.4.6 Тип данных ST (строка символов)

7.4.6.1 Описание

Специализация типа данных ANY

Строковый тип данных используется для текстовых данных, в основном рассчитанных на машинную обработку (например, сортировка, поиск, индексирование и т. д.) или на непосредственный вывод на экран или принтер. Используется для имен, символов, представлений и формальных выражений.

Значение типа данных ST должно содержать как минимум один символ либо иметь причину пустоты nullFlavor.

7.4.6.2 Синтаксис ИСО/МЭК 11404

```
type ST = class (
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    value : characterstring,
    language : characterstring
    translation : Set(ST.NT)
)
```

7.4.6.3 Атрибуты

7.4.6.3.1 value : String: фактическое содержание строки. Обсуждение кодировки символов строк приведено в 6.7.5.

Это пример шаблона обертки примитивного типа данных. Более детальные сведения приведено в 6.3.

7.4.6.3.2 language : Code: человеческий язык содержания. Допустимые коды берутся из документа RFC 3066 организации IETF. Если этот атрибут пуст, то язык может быть идентифицирован откуда-то еще, например, из содержания или из тега языка в кодировке unicode.

В профилях соответствия должны быть определены правила задания языка по умолчанию, применяемые в данной среде использования настоящего стандарта.

Хотя значение атрибута language обычно изменяет интерпретацию текста, содержащийся в нем код языка не меняет значение символов текста.

7.4.6.3.3 translation : Set(ST.NT): перевод, то есть альтернативное представление того же самого содержания на другом языке. Переводы не могут содержать другие переводы.

Переводы должны представлять ту же самую информацию, но на другом языке. Переводы не принимают участия в проверке на равенство и, соответственно, не должны добавлять какую-либо новую семантику к исходному значению.

7.4.6.4 Равенство

Два непустых (nonNull) значения типа ST равны только в том случае, если одинаковы последовательности символов, которые они представляют (то есть они не имеют причины пустоты nullFlavor и значения их атрибутов value равны). Свойство translation исключается из проверки на равенство. Кроме того, из этой проверки исключается свойство языка language, иначе было бы непонятно, как сравнивать значения типа ST, когда язык не указан.

Значение типа данных ED, содержащее простой текст, может быть также равно значению типа данных ST, имеющего то же символьное содержание, при условии выполнения правил, определенных для типа данных ED (см. 7.4.2.4).

7.4.6.5 Инварианты:

- если атрибут value присутствует, он должен содержать по меньшей мере один символ;
- переводы не могут содержать переводы.

Представление инвариантов на языке OCL:

```
inv "вложенные переводы не допускаются": translation->
    forAll(t | t.translation >isEmpty)
inv "содержание, если нет причины пустоты nullFlavor" : isNotNull implies
    (value.oclIsDefined and value.size > 0)
```

7.4.6.6 Операции

7.4.6.6.1 mediaType : String: возвращает значение text/plain.

7.4.6.6.2 size : Integer: число символов в строке.

7.4.6.6.3 concat(other : ST) : ST: конкатенация данного значения со значением other.

7.4.6.6.4 substring(lowe r: INT, upper : INT) : ST: подстрока данного значения, начинающаяся символом с номером lower и завершающая символом с номером upper.

Символы нумеруются от 1 до значения this.size().

Примечание — При извлечении символов из строки может оказаться необходимым скопировать некоторые другие предшествующие символы, образующие необходимый контекст в некоторых кодировках символов.

7.4.6.6.5 toInteger : INT: если содержание строки представляет допустимую запись целого числа, то возвращается его значение как целое число типа INT. Если содержание не представляет допустимую запись целого числа, то возвращается причина пустоты NullFlavor со значением NI (нет информации).

Строка представляет допустимую запись целого числа, если она соответствует формату литерала целого числа, определенному в ИСО/МЭК 11404, или лексическому представлению числа типа integer, определенному в XML-схеме.

7.4.6.6.6 toReal : REAL: если содержание строки представляет допустимую запись числа с плавающей точкой, то возвращается его значение как число типа REAL. Если содержание не представляет допустимую запись числа с плавающей точкой, то возвращается причина пустоты NullFlavor со значением NI (нет информации).

Строка представляет допустимую запись числа с плавающей точкой, если она соответствует формату литерала вещественного числа, определенному в ИСО/МЭК 11404, или лексическому представлению числа типа double, определенному в XML-схеме.

7.4.6.6.7 toTimestamp : TS: если содержание строки представляет допустимую запись штампа даты и времени, то возвращается его значение как значение типа TS. Если содержание не представляет допустимую запись штампа даты и времени, то возвращается причина пустоты NullFlavor со значением NI (нет информации). Строка представляет допустимую запись штампа даты и времени, если она соответствует формату, приведенному в описании типа данных TS.

7.4.6.7 Примеры

```
<example language="en" value="целлюлит левой ноги"/>
```

7.4.7 ST.NT

7.4.7.1 Описание

Тонкость, ограничивающая тип данных ST

Тонкость ST.NT ограничивает тип данных ST таким образом, что переводы недопустимы.

7.4.7.2 Инварианты:

- переводы недопустимы.

Представление инвариантов на языке OCL:

```
inv "переводы недопустимы": translation->isEmpty
```

7.4.8 ST.SIMPLE

7.4.8.1 Описание

Тонкость, ограничивающая тип данных ST.NT

Тонкость ST.SIMPLE ограничивает тип данных ST.NT таким образом, что язык недопустим.

7.4.8.2 Инварианты:

- язык недопустим.

Представление инвариантов на языке OCL:

```
inv "язык недопустим": language.oclIsUndefined
```

7.4.9 Тип данных SC (кодированная строка)

7.4.9.1 Описание

Специализация типа данных ST

Символьная строка, с которой может быть связан необязательный код. Если код присутствует, то обязательно должна присутствовать символьная строка.

Примечание — Этот код часто является местным. Тип данных SC используется в тех случаях, когда кодирование является относительно редким (например, сообщения об ошибках являются в основном текстовыми, и именно текст представляет их основное содержание. Однако иногда сообщения выбираются из каталога заранее подготовленных текстов, на который можно ссылаться с помощью типа данных SC).

Любое непустое значение типаl SC может иметь код, однако код не должен существовать без текста.

Сходство и различия между типами данных SC и CD рассмотрены 7.5.2.2.

7.4.9.2 Синтаксис ИСО/МЭК 11404

```
type SC = class (
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
```

```
    flavorId : Set(characterstring),
    value : characterstring,
    language : characterstring,
    translation : Set(ST.NT),
    code : CD
}
```

7.4.9.3 Атрибуты

7.4.9.4 code : CD: кодированное значение, ассоциируемое со строкой. Если значение пусто или имеет причину пустоты nullFlavor, то никакой код с данной строкой не ассоциируется.

7.4.9.5 Равенство

Определение равенства для типа данных SC то же самое, что и для типа данных ST. Коды исключаются из проверки на равенство, и значения типа SC могут быть равны значениям типа ST (и, следовательно, значениям типа ED при условии выполнении правил, документированных для типа данных ST в 7.4.6.4).

7.4.9.6 Инварианты:

- если код присутствует, то у значения типа должно быть и некоторое содержание SC (следовательно, значение типа SC не должно иметь причину пустоты nullFlavor);
- значение атрибута originalText значения типа SC должно быть пустым (в качестве атрибута originalText выступает значение SC.value).

Представление инвариантов на языке OCL:

```
inv "если нет значения, то нет и кода" : code.isNotNull implies
    isNotNull
inv "у атрибутов типа данных SC нет истории или режима обновления":
    noUpdateOrHistory(code)
inv "нет исходного текста" : code.isNotNull implies
    code.originalText.isNull
```

7.4.9.7 Примеры

```
<example xsi:type="SC" value="Intestinal nematode infection">
<code code="57540006" codeSystem="2.16.840.1.113883.6.96"
codeSystemName="Snomed CT">
<displayName value="Intestinal nematode infection (disorder)"/>
</code>
</example>
<example xsi:type="SC" value="Lung nematode infection"/>
```

Активно используются два примера применения типа данных SC. Если значение типа SC обязательно, то текст требуется, а кодирование не является обязательным. Это нередко удобно при сборе первичных данных, особенно при оказании экстренной медицинской помощи или при снятии боли, когда нет возможности заниматься тщательным выбором понятий. Если понятие известно, оно используется, и его обозначение служит текстом. Если понятие не может быть быстро найдено, то пользователь вводит некоторый текст, который может быть закодирован позже.

7.4.10 SC.NT

7.4.10.1 Описание

Тонкость, ограничивающая тип данных SC

Тонкость SC.NT ограничивает тип данных SC таким образом, что переводы недопустимы.

7.4.10.2 Инварианты:

- переводы недопустимы.

Представление инвариантов на языке OCL:

```
inv "переводы недопустимы": translation->isEmpty
```

7.5 Кодированные типы данных (терминология)

7.5.1 Введение

Эти типы данных обеспечивают использование кодов и терминов из терминологий и классификаций (см. рисунок 4).

7.5.2 Тип данных CD (дескриптор понятия)

7.5.2.1 Описание

Специализация типа данных ANY

Тип данных CD представляет ссылку на понятие, определенное во внешней системе кодирования, терминологии или онтологии. Значение типа CD может содержать простой код, то есть ссылку на понятие, определенное непосредственно в указанной системе кодирования, или выражение, удовлетворяющее некоторому синтаксису, определенному в указанной системе кодирования, и имеющее определенный смысл. Например, понятие «левая нога» может быть представлено как пост-координированный термин, образованный из основного кода «нога» и квалификатора «левый».

Значение типа CD может содержать исходный текст или фразу, послужившую основой для кодирования. Это позволяет обеспечить возможность различных проверок правильности представления понятия.

Значение типа CD может содержать один или несколько переводов в другие системы кодирования. Переводы являются представлениями того же самого понятия в разных системах кодирования. Все они относятся к одному понятию, и только первый экземпляр типа CD может содержать исходный текст. При необходимости можно создать цепочку переводов — какое значение типа CD из какого значения было получено. С каждым значением типа CD может быть также связано обоснование, почему оно представлено.

Значение типа CD, не имеющее причины пустоты nullFlavor, должно иметь атрибут code или непустой (nonNull) атрибут исходного текста originalText. Значение типа CD, имеющее атрибуты code, codeSystem или originalText, но не удовлетворяющее внешним ограничениям на допустимый набор значений, должно иметь атрибут причины пустоты nullFlavor со значением «OTH» (другой).

Атрибуты типа CD обычно связаны внешними ограничениями, наложенными на кодированные понятия, которые могут быть значениями этих атрибутов. Эти ограничения могут быть квалифицированы как «расширяемое» (CWE) или «не расширяемое» (CNE). Если ограничение не расширяемое (CNE), то значение типа CD, у которого нет причины пустоты nullFlavor, должно содержать код, удовлетворяющий этому ограничению. Если ограничение расширяемое (CWE), то значение типа CD, у которого нет причины пустоты nullFlavor, должно содержать или код, существующий в домене, с которым связан атрибут, или код из местной системы кодирования, или атрибут originalText, описывающий понятие. Если код взят из местной системы кодирования, то она должна быть указана в свойстве codeSystem.

Для обоих типов ограничений (CNE и CWE) переводы могут содержать непустые коды из любого источника, если иное не оговорено в модели ограничений.

Ограничения типа CNE могут использоваться и для систем кодирования, в которых определен синтаксис выражений, при условии, что в определениях системы кодирования описана соответствующая поддержка, позволяющая для наборов значений давать полезные заявления о том, как контролировать синтаксис выражений, а также при условии, что используемые механизмы наборов значений тоже имеют соответствующую поддержку.

7.5.2.2 Типы данных CD и SC

Типы данных CD и SC имеют очень схожие структуры. У типа данных CD есть пара атрибутов code:codeSystem с переводами и атрибут исходного текста originalText, имеющий тип данных ED.Text, то есть простой текст, который может быть доступен по значению или по ссылке. У типа данных SC есть строки и код: тип данных CD позволяет строкам быть кодированными. У типа данных SC код не имеет атрибута originalText — его значение приравнено к значению атрибута value типа данных SC. Следовательно, оба типа данных имеют пару code:codeSystem с переводами и исходный текст.

Хотя эти два типа данных имеют одинаковую способность представления кодированного текста и почти одинаковую базовую структуру информации, назначение у них разное. Тип данных CD существует для ссылки на понятие, определенное в некоторой системе кодирования и, возможно, ссылающееся на некоторый поясняющий текст. Когда в контексте использования тип данных CD обязателен, то код должен быть указан, а исходный текст не обязателен, за исключением обсужденного выше случая ограничений CWE, при которых должен быть указан либо код, либо исходный текст. Тип данных SC существует для представления текстового содержания, которому может быть сопоставлен необязательный код. Когда тип данных SC обязателен, текст (который становится исходным текстом) должен быть указан непосредственно, а код всегда не обязателен.

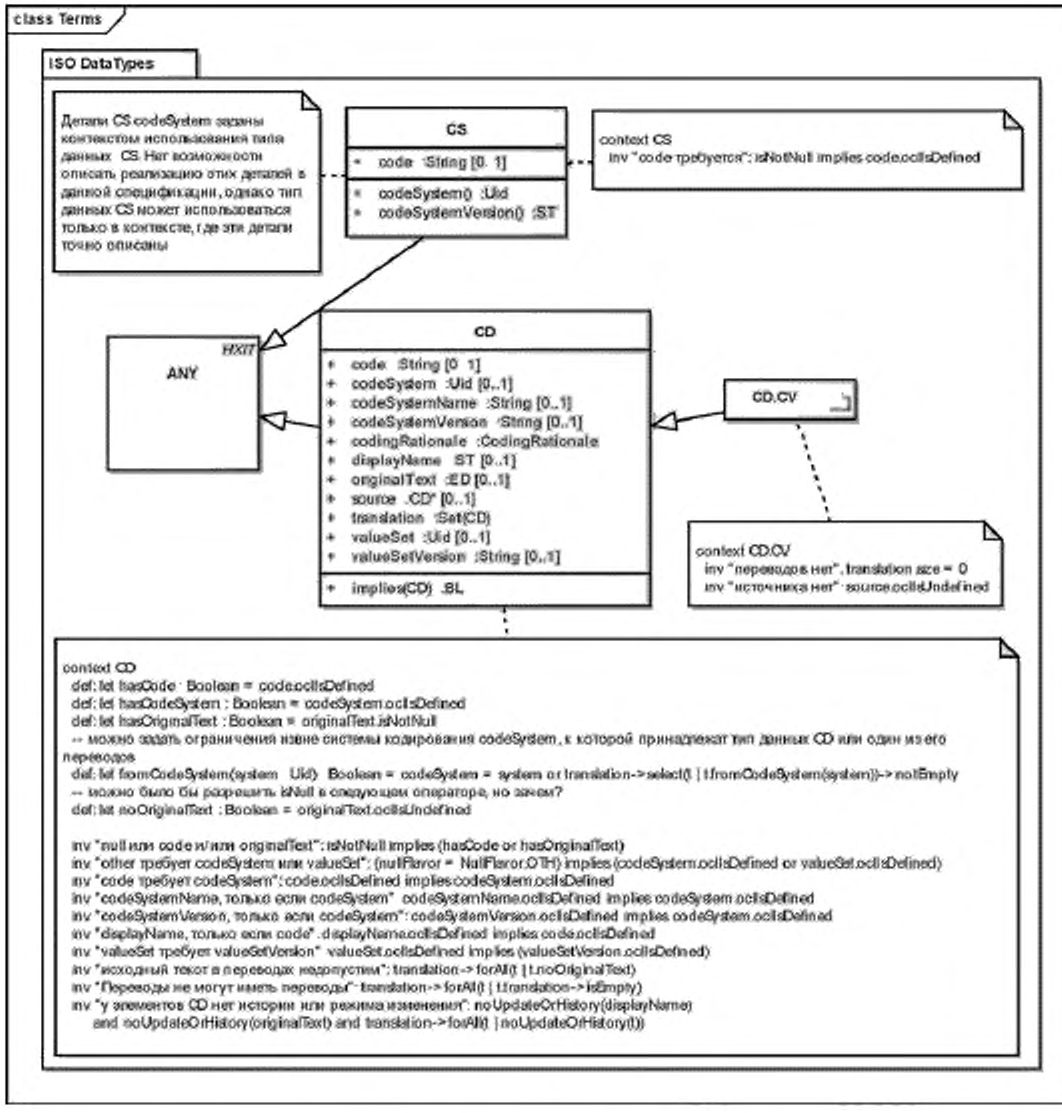


Рисунок 4 — Кодированные типы данных

Когда очевидно, какой аспект кодированного текста обязателен — код или текст, то становится очевидным, использовать тип данных CD или SC. С другой стороны, если ни один из аспектов не обязателен либо обязательны оба, то выбор из этих типов данных становится не очевиден.

Примечание — Если оба обязательны, то каждый из этих классов может быть ограничен требованием использовать оба аспекта, представленным на некотором формальном языке описания ограничений, например, OCL.

Обычно в случаях, когда текст кодируется постфактум, очевидным выбором является тип данных CD, поскольку он позволяет сослаться на существующий текст. В других случаях правильный выбор между типами SC и CD неоднозначен. При выборе пользователям рекомендуется рассмотреть иерархию специализации, которая может диктовать конкретный выбор (может оказаться полезным, что тип данных SC является специализацией типа данных ST, а может и нет), а также рассмотреть ограничения,

накладываемые на тип данных контекстом его использования; они также могут диктовать конкретный выбор.

В некоторых ситуациях выбор типа так и остается неоднозначным. При наличии сомнений разработчикам рекомендуется выбрать тип данных CD по умолчанию.

7.5.2.3 Синтаксис ИСО/МЭК 11404

```
type CD = class {
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    code : characterstring,
    codeSystem : characterstring,
    codeSystemName : characterstring,
    codeSystemVersion : characterstring,
    valueSet : characterstring,
    valueSetVersion : characterstring,
    displayName : ST,
    originalText : ED,
    codingRationale : CodingRationale,
    translation : Set(CD),
    source : CD
}
```

7.5.2.4 Атрибуты

7.5.2.4.1 code : String: простой символьный код, определенный в системе кодирования, или выражение, соответствующее синтаксису, определенному в системе кодирования, описывающей это понятие.

Значение атрибута code должно точно совпадать с кодом или выражением, определенным в системе кодирования. Если в ней определен код или выражение, включающее в себя пробельный символ, то значение атрибута code также должно содержать этот пробельный символ. Выражение может использоваться только в том случае, если синтаксис выражения определен либо в системе кодирования, либо в общем синтаксисе, приемлемом для системы кодирования. Может быть определена такая система кодирования, которая определяет только синтаксис выражений, а элементы выражений берутся из других систем кодирования.

Интерпретирующая система по своему усмотрению может решать, проверять ли значение кода на предмет, является ли оно выражением, и интерпретировать это выражение либо трактовать выражение как код. В некоторых случаях такое решение может оказаться затруднительным или неоднозначным. Обычно эти затруднения возникают, когда система кодирования определяет язык выражений, а затем определяет пре-координированные понятия с кодами, совпадающими с выражениями, как это имеет место в системе единиц измерений UCUM. В других случаях безопасно трактовать выражение как простой код. Однако никаких гарантий безопасности нет: надо всегда ознакомиться с описаниями системы кодирования, чтобы определить, как обрабатывать потенциально существующие выражения.

7.5.2.4.2 codeSystem: Uid . идентификатор системы кодирования, в которой определено значение атрибута code, или, если код не найден, идентификатор системы кодирования, в которой код не был найден.

Для идентификации систем кодирования должны использоваться уникальные идентификаторы, обеспечивающие однозначные ссылки на стандартные системы кодирования и другие местные системы. Если или организация ИСО, или организация HL7 уже присвоили такой идентификатор системе кодирования, то должен использоваться этот идентификатор. В остальных случаях в качестве глобально уникального идентификатора местной системы кодирования должен использоваться либо подходящий объектный идентификатор ИСО (ОИД), либо универсально уникальный идентификатор UUID.

Значение типа CD, имеющее атрибут code, должно иметь также и атрибут codeSystem, указывающий систему кодирования, в которой определено значение атрибута code.

7.5.2.4.3 codeSystemName : String: общее имя системы кодирования.

Имя системы кодирования не используется для вычислений. Оно не должно модифицировать значение свойства codeSystem и не может существовать без значения свойства codeSystem.

Элементы обработки информации, объявляющие непосредственное или косвенное соответствие настоящему стандарту, не должны функционально полагаться на значение свойства codeSystemName. Кроме того, это свойство может быть в них не реализовано, но при этом они не должны отклонять экземпляры типа CD, если в них присутствует свойство codeSystemName.

Примечание — Имя системы кодирования предназначено для облегчения интерпретации человеком значений свойств code и codeSystem.

7.5.2.4.4 codeSystemVersion : String: если применим, дескриптор версии, специально определенный для данной системы кодирования.

Различные версии одной и той же системы кодирования должны быть совместимыми. По определению код должен иметь тот же самый смысл во всех версиях системы кодирования. При переходе к более новой версии коды могут быть признаны устаревшими, но они никогда не должны удаляться или повторно использоваться для другого понятия. Если смысловое определение кода меняется, оно тем не менее должно быть совместимым (равным) в разных версиях системы кодирования.

Если система кодирования изменена несовместимым образом, то она представляет собой другую систему кодирования, а не другую версию, как бы издатель это ни называл. Например, издатель классификаций МКБ-9 и МКБ-10 назвал эти системы кодирования «9-м пересмотром» и «10-м пересмотром». Однако классификация МКБ-10 представляет собой полное изменение кодов МКБ и не является обратно совместимой. Поэтому в целях настоящего стандарта типы данных классификации МКБ-9 и МКБ-10 рассматриваются как разные системы кодирования, а не как разные версии одной системы кодирования. Напротив, когда версия «1.0j» системы кодирования LOINC была обновлена до версии «1.0k», она должна была рассматриваться как другая версия, поскольку версии LOINC обратно совместимы.

7.5.2.4.5 valueSet : Uid: набор значений, использованный при кодировании данного значения типа CD.

Ссылки на наборы значений должны представлять собой имена идентификаторов, позволяющие однозначно определить данный набор значений.

Если или организация ИСО, или организация HL7 уже присвоили такое имя идентификатора набору значений, то следует использоваться это имя.

Во многих случаях значение типа CD создается из набора значений — либо оно выбирается из набора значений, либо задается не входящим в этот набор, и тогда значение типа CD является исключительным с причиной пустоты NullFlavor.OTH (другое). Если код не выбран, то в общем случае нельзя задавать систему кодирования, из которой выбран код, поскольку набор значений может не совпадать с системой кодирования (он может быть подмножеством системы кодирования или включать в себя дополнительные термины из других систем кодирования); вместо этого надо указывать набор значений. Кроме того, известны некоторые сценарии, в которых набор значений, предложенный пользователю или системе при выборе кода, влияет на интерпретацию этого кода.

Если код задан, то его смысловое значение должно браться из определения кода в системе кодирования. Оно не должно зависеть от набора данных. От элементов обработки информации, объявляющих непосредственное или косвенное соответствие настоящему стандарту, не должна требоваться особая интерпретация кода в зависимости от набора значений valueSet, и они не должны отклонять экземпляр из-за его присутствия или отсутствия в конкретном наборе значений.

7.5.2.4.6 valueSetVersion : String: версия набора значений valueSet, в которой код был найден.

Атрибут valueSetVersion должен присутствовать, если присутствует атрибут valueSet. В противном случае он должен быть пустым. Значение атрибута valueSetVersion должно идентифицировать конкретную версию набора значений в соответствии с правилами, определенными издателем этого набора.

Обычно рекомендуется, чтобы издатели наборов значений указывали, что версия идентифицируется датой и временем публикации и что процесс публикации позволяет точно интерпретировать эти дату и время.

7.5.2.4.7 displayName : ST: имя, заголовок или представление кода либо выражения в том виде, как оно указано в системе кодирования.

Если атрибут displayName присутствует, то он должен содержать человекочитаемое представление понятия, определенного в системе кодирования на момент ввода данных. Оно должно удовлетворять

всем правилам, определенным в системе кодирования, указанной в атрибуте codeSystem. Если в ней не определено человекочитаемое представление кода или выражения, то атрибут displayName должен отсутствовать. Атрибут displayName включен как для удобства интерпретации человеком значения кода, так и для документирования имени, используемого для изображения понятия пользователю. Оно не имеет функционального значения, не может существовать без кода и никогда не должно модифицировать смысл кода. Это имя может отсутствовать, если код представляет собой выражение, которому человекочитаемое представление не присвоено или не может быть выведено. Элементы обработки информации, объявляющие непосредственное или косвенное соответствие настоящему стандарту, могут не реализовать свойство displayName, но при этом они не должны отклонять экземпляр из-за присутствия в нем этого свойства.

Человекочитаемые имена, предусмотренные для кодов, не могут менять смысл кодированного значения. Поэтому они не должны предоставляться пользователю прикладной системы-получателя, пока не будет уверенности в том, что такое имя адекватно отражает понятие, соответствующее кодированному значению. При коммуникациях нельзя просто полагаться на имя кодированного значения. Основная цель этого имени в обеспечении возможности отладки реализаций.

7.5.2.4.8 originalText : ED: текст, видимый или выбранный пользователем, который ввел данные, представляющие ему необходимый смысл.

Примечание — Местные соглашения могут влиять на то, что требуется для представления этого исходного текста.

Атрибут исходного текста originalText может быть использован в структурированном интерфейсе пользователя для выбора того, что пользователь видит как представление кода в форме ввода данных, или в ситуации, когда пользователь диктует текст или вводит его непосредственно в форму, это текст, введенный или произнесенный пользователем.

Вполне допустимо использовать тип данных CD для хранения только текста, введенного или произнесенного пользователем. В этом случае исходный текст будет существовать без кода. В случае, когда код был присвоен информации через какое-то время после ее появления, значением атрибута originalText должны быть текст или фраза, использованные в качестве основы для кодирования.

Детали связи значения originalText.reference с различными артефактами медицинской информации (например, документом и кодированным результатом) не входят в область применения настоящего стандарта и могут быть далее рассмотрены в спецификациях, разработанных на его основе.

Исходный текст должен представлять собой извлечение из оригинального источника, а не точную копию его содержания или указатель на этот источник. Поэтому исходный текст должен быть представлен в неформатированном виде. В некоторых случаях, когда контекст использования исходного текста точно описан, значение атрибута originalText может быть ссылкой на некоторый другой текстовый артефакт, для которого точно описаны рамки применения.

Значения типа CD могут иметь непустое свойство исходного текста, даже если свойство code имеет пустое значение. Данное значение считается исключительным. В этом случае свойство originalText является именем или описанием понятия, которое не было закодировано. Такие значения типа CD могут также содержать переводы translation.

Подобные переводы обеспечивают непосредственное кодирование понятия, описанного в свойстве originalText. Это свойство представляет исходный текст описания этого понятия. Переводы не должны содержать собственные атрибуты исходного текста.

7.5.2.4.9 translation : Set(CD): множество других значений типа CD, которые представляют переводы данного значения типа CD в эквивалентные коды внутри той же системы кодирования или в соответствующие понятия, описанные в других системах кодирования.

Переводы являются квазисинонимами одного понятия реального мира. Предполагается, что каждый перевод в этом множестве выражает то же самое понятие «другими словами». Однако между двумя структурно различными системами кодирования редко существует точная синонимия. Поэтому не все преобразования будут абсолютно точными.

Переводы не должны содержать другие переводы. Корневое значение типа CD обладает одним множеством переводов, в котором перечислены все переводы. Главным переводом обычно является тот, который наилучшим образом удовлетворяет критериям соответствия типу данных CD. Из выбора корневого значения не должно следовать никаких выводов о происхождении перевода. Для прослеживания происхождения следует использовать атрибуты codingRationale и source.

В отсутствие модели ограничений домена значений типа данных CD любой из переводов может быть корневым значением типа CD. Если существует модель, ограничивающая домен значений типа CD, и существует перевод, который удовлетворяет этим ограничениям, то этот перевод рекомендуется выбрать в качестве корневого значения типа CD. Если существует модель, ограничивающая домен значений типа CD, но при этом ни один из переводов не удовлетворяет этим ограничениям, то корневым значением может быть любой перевод при условии, что ему присвоена причина пустоты nullFlavor. Может использоваться альтернативный вариант, при котором ни один из переводов не объявляется корневым значением, самому значению типа CD присваивается одна из причин пустоты, описанных в словарном домене NullFlavor, а все переводы помещаются в свойство translation корня.

7.5.2.4.10 codingRationale : CodingRationale: причина, по которой конкретное значение типа CD было представлено или как корневое понятие, или как один из переводов.

Если значение этого атрибута присутствует, то оно должно быть взято из перечисления, описанного в таблице 11. Это перечисление образовано из системы кодирования HL7 CodingRationale.

Таблица 11 — Перечисление CodingRationale, ОИД: 2.16.840.1.113883.5.1074

Уровень	Код	Имя	Определение
1	O	Original	Оригинально произведенный код
1	P	Post-coded	Кодированный постфактум из источника свободного текста
	R	Required	Требуется спецификацией, описывающей использование кодированного понятия. Точная форма требования здесь не указана; требование может вытекать непосредственно из спецификации или может опосредованно вытекать из других инструментов соответствия. Поскольку одновременно может существовать несколько разных требований, то может понадобиться более одного кода в комплексе CD
	OR	Original and required	Оригинально произведенный код требуется спецификацией, описывающей использование кодированного понятия
	PR	Post-coded and required	Кодированный постфактум из источника свободного текста требуется спецификацией, описывающей использование кодированного понятия

Синтаксис ИСО/МЭК 11404 атрибута codingRationale:

```
type CodingRationale = enumeration (O, P, R, OR, PR)
```

Код считается присвоенным постфактум, если пользователь не задал код, когда впервые ввел данные. Атрибут codingRationale не должен рассматриваться в качестве признака контроля качества процессов кодирования или перевода.

Атрибут code требуется, если он присутствует в экземпляре для удовлетворения некоторых ограничений, наложенных на экземпляр контекстом использования. От элементов обработки информации не должно требоваться помечать конкретный перевод как требуемый, даже если это вытекает из контекста использования, но они могут это делать. Элементы обработки информации не должны отклонять экземпляры из-за наличия или отсутствия признака codingRationale.

7.5.2.4.11 source : CD: если данное значение типа CD было создано как перевод другого значения типа CD, то атрибут source содержит ссылку на то значение типа CD, которое послужило источником для данного перевода.

Это свойство является ссылкой. Источник, на который оно указывает, должен быть представлен из числа корневого значения типа CD и его переводов; другими словами, оно должно быть другим представлением того же понятия в том же атрибуте.

Значение типа CD состоит из единственного корневого кода и множества переводов, которые не имеют других переводов. Используя свойство codingRationale, отправитель может указать, какой код является оригинальным. Бывают обстоятельства, при которых полезно знать, какое значение типа CD является

переводом какого значения типа CD. С помощью атрибута source можно представить последовательность от одного перевода к другому. Каждый элемент множества переводов был получен из исходного значения типа CD. Однако каждый перевод также может иметь переводы. Таким образом, когда код переводится несколько раз, будет сохранена информация о том, какой код послужил входным значением для какого перевода.

7.5.2.5 Равенство

Равенство двух значений типа CD определяется исключительно на основе сравнения их атрибутов code и codeSystem. Атрибуты codeSystemVersion, originalText, codingRationale, source, информация о наборе значений и переводах не участвуют в проверке на равенство. Значения, имеющие причину пустоты nullflavor, не равны, даже если у них одинаковые коды причины пустоты или значения атрибутов исходного текста.

Равенство основано на литеральных значениях атрибутов code и codeSystem. Элементы обработки информации не должны рассматривать семантическое значение пары code+codeSystem для определения того, идентифицирует ли она то же самое понятие.

Примечания

1 Это, к примеру, означает для номенклатуры SNOMED, что две изоморфные формы одного и того же выражения не будут равны. При реализации следует тщательно выбирать литеральные формы представления.

2 Значения типа CD могут быть также равны значениям типа CS. Детальная информация приведена в 7.8.6.4.

7.5.2.6 Инварианты:

- если значение не пусто, то либо атрибут code, либо атрибут originalText должны иметь значение;
- если атрибут code имеет значение, то атрибут codeSystem тоже должен иметь значение;
- присутствие атрибута valueSet требует присутствия атрибута valueSetVersion;
- атрибут codeSystemName может иметь значение только в том случае, если атрибут codeSystem имеет значение;
- атрибут codeSystemVersion может иметь значение только в том случае, если атрибут codeSystem имеет значение;
- атрибут displayName может иметь значение только в том случае, если атрибут code имеет значение;
- у переводов не может быть исходного текста;
- переводы не могут иметь переводы.

Представление инвариантов на языке OCL:

```
def: let hasCode : Boolean = code.oclIsDefined
def: let hasCodeSystem : Boolean = codeSystem.oclIsDefined
def: let hasOriginalText : Boolean = originalText.isNotNull
  codeSystem = system or translation >select(t |
  t.fromCodeSystem(system)) >notEmpty
```

Примечание — Определена система кодирования fromCodeSystem, являющая источником, поэтому можно задать ограничения извне системы кодирования codeSystem, к которой принадлежат тип данных CD или один из его переводов, например, inv: code.fromCodeSystem(«2.16.840.1.113883.6.42»).

```
def: let noOriginalText : Boolean = originalText.oclIsUndefined
inv "null или code и/или originalText":
    sNotNull implies (hasCode or hasOriginalText)
inv "other требует codeSystem или valueSet":
    (nullFlavor = NullFlavor.OTH) implies
    (codeSystem.oclIsDefined or valueSet.oclIsDefined)
inv "code требует codeSystem": code.oclIsDefined
    implies codeSystem.oclIsDefined
inv "codeSystemName, только если codeSystem":
    codeSystemName.oclIsDefined implies codeSystem.oclIsDefined
inv "codeSystemVersion, только если codeSystem":
    codeSystemVersion.oclIsDefined implies codeSystem.oclIsDefined
inv „displayName, только если code": displayName.oclIsDefined
    implies code.oclIsDefined
```

```
inv „valueSet требует valueSetVersion":
    valueSet.oclIsDefined implies (valueSetVersion.oclIsDefined)
inv "исходный текст в переводах недопустим":
    translation >forAll(t   t.noOriginalText)
inv "Переводы не могут иметь переводы":
    translation >forAll(t   t.translation->isEmpty)
inv "у элементов CD нет истории или режима изменения":
    noUpdateOrHistory(displayName) and
    noUpdateOrHistory(originalText)
    and translation >forAll(t | noUpdateOrHistory(t))
```

7.5.2.7 Операции

7.5.2.7.1 implies(other : CD) : BL: является следствием, то есть имеет значение true, если данная пара code+codeSystem специализация другой пары code+codeSystem или имеет то же самое значение.

Примечания

1 В номенклатуре SNOMED, к примеру, две изоморфные формы одного и того же выражения являются следствием друг друга.

2 Для вычисления этого отношения может понадобиться терминологическая служба.

7.5.2.8 Примеры

7.5.2.8.1 Примеры кодов МКБ

Простым примером кода может служить шифр головной боли в классификации МКБ-9, а именно «784.0».

```
<example code="784.0" codeSystem="2..6.840.1.113883.6.42"
    codeSystemName="ICD 9">
    <displayName value="Головная боль, БДУ"/>
    <originalText value="головная боль"/>
</example>
```

Возможным эквивалентом этого кода в МКБ-10 является «G44.1» (классификации МКБ-10 слегка отличаются).

```
<example code="G44.1" codeSystem="2.16.840.1.113883.6.3"
    codeSystemName="ICD-10">
    <displayName value="Сосудистая головная боль, не классифицированная в других
рубриках"/>
    <originalText value="сосудистая головная боль"/>
</example>
```

7.5.2.8.2 Примеры невозможности кодирования

Достаточно часто при использовании типа данных CD фактически используемое понятие не может быть правильно представлено в конкретной системе кодирования. Обычно это происходит, когда ожидается, что понятие присутствует в конкретной системе кодирования. Для целей настоящих примеров предполагается, что все они относятся к результату исследования типа CD, принадлежащего к полному набору значений Snomed-CT (в примере использован ОИД набора значений, равный 2.16.840.1.113883.19.11.1 по состоянию на 11 июня 2007 года, фактический ОИД системы кодирования SNOMED-CT равен 2.16.840.1.113883.6.96). Важно отметить следующее: корневой ОИД 2.16.840.1.113883.19 определен рабочей группой HL7 только для использования в примерах, и те ОИД, которые принадлежат этой ветви, никогда не должны использоваться в реальных экземплярах. Вместо ОИД, использованных в приведенных ниже примерах, в реальных экземплярах надо использовать пространства имен ОИД 2.16.840.1.113883.6, 2.16.840.1.113883.5 и 2.16.840.1.113883.11.

В простейшем случае значение типа CD вообще не представлено в экземпляре или представляется как отсутствие информации:

```
<value nullFlavor="NI"/>
```

Это представление, однако, не очень полезное; часто система-источник обладает большей информацией и целесообразно передать ее системе-получателю, помечая как неполную:

```
<value nullFlavor="OTH" codeSystem="2.16.840.1.113883.6.96"/>
```

Возможен и такой вариант:

```
<value nullFlavor="OTH" valueSet="2.16.840.1.113883.19.11.1"
valueSetVersion="20070711"/>
```

В этом примере версия набора значений valueSetVersion представляет собой штамп даты и времени ближайшего дня. Фактическое значение, разрешенное для данного атрибута, зависит от определения набора значений. В данном случае предполагается, что в определении набора значений 2.16.840.1.113883.19.11.1 указано, что идентификатором версии служит день официальной публикации версии набора значений ее владельцем.

Оба приведенных выше примера означают, что понятие не может быть закодировано в SNOMED. Еще полезнее передать некоторую специфичную информацию об этом понятии, даже если оно не может быть представлено в SNOMED:

```
<value nullFlavor="OTH" codeSystem="2.16.840.1.113883.6.96">
    <originalText value="Ожог уха утюгом. Ожог другого уха при звонке в скорую
    помощь"/>
</value>
```

Возможна ситуация, при которой содержание было сначала закодировано в некоторой другой системе кодирования, а система-источник не смогла закодировать его в SNOMED. В этом случае возможны два варианта. Первый используется, если кодирование в SNOMED помечено квалификатором CWE, то есть допускаются местные расширения:

```
<value code="ожог" codeSystem="2.16.840.1.113883.19.5.2">
    <originalText value="Ожог уха утюгом. Ожог другого уха при звонке в скорую
    помощь"/>
</value>
```

Если кодирование имеет квалификатор CWE, местные расширения разрешены и система-источник может попросту использовать свое собственное значение атрибута codeSystem (идентифицирующее систему кодирования с ОИД «2.16.840.1.113883.19.5.2», предназначенную для примеров) для расширения другой системы кодирования. В действительности система-источник может также использовать код из другой широко распространенной системы кодирования, например, МКБ-9. Если бы в МКБ-9 был код «А10.1», означающий то же самое понятие, то следующее представление значения типа CD было бы правильным:

```
<value code="A10.1" codeSystem="2.16.840.1.113883.6.42">
    <originalText value="Ожог уха утюгом. Ожог другого уха при звонке в скорую
    помощь"/>
</value>
```

Если же использование набора значений SNOMED CT помечено квалификатором CNE, то код должен быть взят только из SNOMED. В этом случае та же самая информация, которая приведена выше, должна быть передана иным образом:

```
<value nullFlavor="OTH" codeSystem="2.16.840.1.113883.6.96">
    <originalText value="Ожог уха утюгом. Ожог другого уха при звонке в скорую
    помощь"/>
    <translation code="ожог" codeSystem="2.16.840.1.113883.19.5.2">
</value>
```

Теперь код точно помечен причиной пустоты OTH (другой): код не может быть взят из SNOMED CT, но предусмотрен перевод понятия в другой системе кодирования. Хотя в данном случае это избыточно, система-источник могла бы следующим образом указать с помощью атрибута source, из какой системы получен какой перевод:

```
<value nullFlavor="OTH" codeSystem="2.16.840.1.113883.6.96">
    <originalText value="Ожог уха утюгом. Ожог другого уха при звонке в скорую
    помощь"/>
    <translation id="s1" code="ожог"
    codeSystem="2.16.840.1.113883.19.5.2">
    <source xref="s1"/>
</value>
```

Во всех этих примерах предполагалось, что атрибут связан с фиктивным набором данных 2.16.840.1.113883.19.11.1, все значения которого взяты из SNOMED CT. Если бы набор значений был расширен кодами LOINC, то было бы неправильно следующим образом указывать невозможность кодирования:

```
<value nullFlavor="OTH" codeSystem="2.16.840.1.113883.6.96"/>
```

поскольку неверно утверждать, что понятие не может быть закодировано из-за того, что соответствующего кода нет в SNOMED CT; оно не может быть закодировано, поскольку соответствующий код не найден ни в SNOMED CT, ни в LOINC. Поэтому правильнее указать невозможность кодирования в форме ссылки на набор значений:

```
<value nullFlavor="OTH" valueSet="2.16.840.1.113883.19.11.1" valueSetVer
sion="20070711"/>
```

7.5.2.8.3 Примеры выражений

Выражения обычно возникают при использовании сложных медицинских терминологий, например, SNOMED. К примеру, в SNOMED CT определены понятие целлюлита «cellulitis (disorder)» (128045006), атрибута места исследования «finding site» (363698007) и понятие структуры ноги «foot structure (body structure)» (56459004). Номенклатура SNOMED CT позволяет составить из этих кодов следующее предложение, означающее целлюлит ноги.

```
128045006|cellulitis (disorder)|:{363698007|finding site|=56459004|foot
structure|}
```

Полная форма значения типа CD для этого выражения будет иметь следующий вид:

```
<value code="128045006:{363698007=56459004}"
codeSystem="2.16.840.1.113883.6.42" codeSystemName="Snomed CT">
<originalText value="Целлюлит ноги"/>
</value>
```

Язык композиционных выражений, используемый в номенклатуре SNOMED, позволяет включать в выражение не только коды, но и термины, как это показано в первом примере. Это делает выражение более читаемым для человека и таким образом используется в настоящем подразделе для автономных выражений. Однако включение терминов в выражение является не обязательным и ничем не улучшает возможности компьютерной обработки, более того, добавляет ненужные новые сложности, например, при проверке на равенство. По этой причине не рекомендуется включать термины в выражения, используемые в экземплярах типа данных CD, и в настоящем стандарте не будет примеров такого включения. В описание наборов значений могут включаться правила, регулирующие включение или отсутствие терминов в выражениях.

Язык композиционных выражений, используемый в номенклатуре SNOMED, в настоящее время находится в стадии рецензирования. Его описание можно найти на сайте организации IHTSDO

(International Health Terminology Standards Development Organization). Следующие два примера основаны на версии SNOMED CT Core Edition 2007-01-31.

В первом примере используется код SNOMED для понятия «fracture of left tibia» (перелом левой большеберцовой кости). Он демонстрирует аспекты группировки и вложения.

```
31978002|fracture of tibia|: 272741003|laterality|=7771000|left
```

Строго говоря (в нормальной форме), «fracture of left tibia» означает не «left fracture of a tibia bone» (левый перелом большеберцовой кости), а «fracture of the left tibia bone» (перелом левой большеберцовой кости), то есть квалификатор «left» (левый) применяется к кости, а не к перелому. Кроме того, в этом примере перелом и кость сгруппированы; это может показаться не существенным, но потенциально имеет значение для сочетанных переломов, когда для разных костей может быть указана разная морфология. Альтернативное представление этого же понятия имеет следующий вид:

```
64572001|disease|:{116676008|associated morphology|=72704001|fracture|,
363698007|finding site|=(12611008|bone structure of
tibia|:272741003|laterality|=7771000|left|)}
```

Второй пример демонстрирует более сложную группировку и вложенную структуру. Выражение SNOMED CT для понятия «past history of fracture of left tibia» (история лечения перелома левой большеберцовой кости) включает в себя вложенность даже в ее простейшей форме, поскольку понятие стороны применяется не к истории лечения, а к нарушению здоровья:

```
417662000|past history of clinical finding|:246090004|associated finding|=
(31978002|fracture of tibia|: 272741003|laterality|=7771000|left|)
```

Альтернативное представление использует еще большую вложенность:

```
243796009|situation with explicit context|:246090004|associated finding|=
(64572001|disease|:{116676008|associated morphology|=72704001|fracture|,
363698007|finding site|=(12611008|bone structure of tibia|:
272741003|laterality|=7771000|left|)}),408729009|finding context|=
410515003|known present|,408731000|temporal context|=410513005|past|,
408732007|subject relationship context|=410604004|subject of record|
```

Эти представления приведены как примеры синтаксиса выражений в номенклатуре SNOMED. Полное обсуждение достоинств различных форм, их взаимосвязей и того, как ими пользоваться, можно найти в упомянутом выше описании языка композиционных выражений.

Важно иметь в виду, что синтаксис выражений и семантические правила определены в системе кодирования. Например, в SNOMED CT определен набор квалифицирующих атрибутов, и только исследования (findings) и нарушения (disorders) могут быть квалифицированы значением атрибута «finding site» (место исследования). В типе данных CD не предусмотрена нормализация композиционных выражений, поэтому существует возможность создания неоднозначно трактуемых выражений. Пользователи должны отдавать себе отчет в том, что при создании композиционных выражений, включаемых в значение типа CD, для однозначного представления данных необходимо использовать дополнительные ограничения. В противном случае они рискуют отсутствием возможности получения полного набора всех записей, удовлетворяющих условиям конкретного запроса.

Классификация МКБ-10 допускает двойное кодирование (см., например, 3.1.3 руководства по МКБ-10). Хотя в классификации МКБ-10 описаны точные правила, являющиеся семантической основой двойного кодирования, в ней не приводится конкретная литеральная форма выражения, пригодная для использования в типе данных CD. В связи с этим рабочая группа HL7 самостоятельно определила подходящую литеральную форму выражения и присвоила ей ОИД 2.16.840.1.113883.6.260. В этой системе кодирования указано, что два кода МКБ-10 разделяются пробелом:

```
value code="J21.8 B95.6" codeSystem="2.16.840.1.113883.6.260"
codeSystemName="ICD-10 Dual Code Expression">
```

```
<originalText value="Острый бронхиолит, вызванный Staphylococcus aureus"/>
</value>
```

Шифр J21.8 в МКБ-10 соответствует понятию «Острый бронхиолит, вызванный другими уточненными агентами», а шифр B95.6 — понятию «Staphylococcus aureus как причина болезней, классифицированных в других рубриках».

Выражения возникают также в системе единиц измерения UCUM. Поскольку эта система стабильна, выражения кодов из системы UCUM обычно встречаются в атрибуте единиц измерения, определенном в типе данных PQ (см. 7.8.9.3.2). Хотя этот атрибут имеет тип данных CS, все равно это служит примером указания выражения в коде (тип данных CS может быть преобразован в тип данных CD с помощью явного указания атрибута системы кодирования codeSystem). Ниже приведено простое выражение из системы UCUM, которое фактически является прямой ссылкой на простое понятие, определенное в системе UCUM:

```
<value xsi:type="PQ" value="1" unit="g">
```

Единица g (г) непосредственно определена в системе UCUM. Типичное выражение, соответствующее системе UCUM, является более сложным:

```
<value xsi:type="PQ" value="1" unit="mmol/l">
```

Концентрация аналита равна 1 mmol/l (ммоль/л); здесь mmol/l — выражение, соответствующее системе UCUM, как и следующее более простое выражение:

```
<value xsi:type="PQ" value="1" unit="mmol">
```

Количество аналита равно 1 mmol (ммоль); mmol также является выражением, соответствующим системе UCUM, а именно, сочетанием приставки m (милли) и единицы mol (моль).

7.5.3 Тонкость CD.CV (кодированное значение)

7.5.3.1 Описание

Ограничение типа данных CD

Кодированные данные, которые могут содержать только код, система кодирования и необязательные атрибуты изображаемого имени и исходного текста. Используется только как тип свойств других типов данных.

Тип данных CV используется, когда в соответствующем сценарии требуется передать только простое кодированное значение. Поэтому он не должен использоваться в ситуации, когда желательно иметь несколько кодов одного и того же понятия или когда требуется мигрировать в новую систему кодирования.

7.5.3.2 Инварианты:

- переводы недопустимы;
- источник недопустим.

Представление инвариантов на языке OCL:

```
inv "переводы недопустимы": translation.size = 0
inv "источник недопустим": source.oclIsUndefined
```

7.5.4 Тип данных CS (кодированное простое значение)

7.5.4.1 Описание

Специализация типа данных ANY

Кодированные данные в простейшей форме, где только код не предопределен.

Система кодирования и версия системы кодирования подразумеваются и фиксированы в контексте использования значения типа CS.

Вследствие очень ограниченной функциональности тип данных CS должен использоваться только для простых структурных атрибутов со строго контролируемыми и стабильными терминологиями, в которых:

- все коды берутся из одной системы кодирования;
- коды, соответствующие запрещенным понятиям, повторно не используются;
- процедура публикации и свойства расширяемости системы кодирования хорошо описаны и понятны.

7.5.4.2 Синтаксис ИСО/МЭК 11404

```
type CS = class {
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    code : characterstring
}
```

7.5.4.3 Атрибуты

7.5.4.3.1 code : String: простой символьный код, определенный в системе кодирования. Если значение кода является пустой строкой или пустым значением, то в системе кодирования нет кода, представляющего понятие.

Коды могут содержать только символы, являющиеся буквой, цифрой или одним из знаков '.', '-', '_' или ':'. В системах кодирования, используемых для значений типа CS, не должны определяться символы кода или синтаксис выражений, содержащих пробельные символы или другие символы, отличающиеся от приведенных выше.

7.5.4.4 Равенство

Равенство двух значений типа CS определяется исключительно на основе явно заданного кода и неявно используемой системы кодирования codeSystem.

Значение версии системы кодирования codeSystemVersion не участвует в проверке на равенство. Значения, имеющие причину пустоты nullFlavor, не равны, даже если коды причины пустоты nullFlavor у них одинаковые.

Равенство основано на литеральных значениях атрибутов code и codeSystem. Элементы обработки информации не должны рассматривать семантическое значение пары code+codeSystem для определения того, идентифицирует ли она то же самое понятие.

Примечание — Значения типа CS могут быть равны значениям типа CD, если у них одинаковы коды и системы кодирования.

7.5.4.5 Инварианты:

- если у значения нет причины пустоты nullFlavor, то код должен присутствовать.

Представление инвариантов на языке OCL:

```
inv "code is required": isNotNull implies
    code.oclIsDefined
```

7.5.4.6 Операции

7.5.4.6.1 codeSystem() : Uid: хотя у типа данных CS нет атрибута codeSystem, всегда должна существовать система кодирования, определяемая контекстом использования типа данных CS. Эта операция возвращает значение системы кодирования codeSystem, заданной контекстом.

Эта операция всегда должна возвращать допустимое значение системы кодирования codeSystem, даже в том случае, когда значение типа CS имеет причину пустоты nullFlavor, поскольку система кодирования определяется контекстом использования.

7.5.4.6.2 codeSystemVersion() : String: хотя у типа данных CS нет атрибута codeSystemVersion, тем не менее может существовать версия системы кодирования, определяемая контекстом использования типа данных CS.

Эта операция возвращает значение версии системы кодирования codeSystemVersion, если оно известно из контекста.

7.5.4.7 Примеры

```
<code xsi:type="CS" code="NS"/>
```

Простой код NS в подразумеваемой системе кодирования.

7.6 Типы данных идентификации и местонахождения

7.6.1 Введение

Эти типы данных используются при идентификации объектов, записей и предметов, в особенности для указания адресов URL, идентификаторов URI и телекоммуникационных адресов (см. рисунок 5).

7.6.2 Тип данных TEL (телекоммуникационный адрес)

7.6.2.1 Описание

Специализация типа данных ANY

Определение: указатель ресурса, идентифицируемый унифицированным идентификатором ресурсов (URI), например веб-страница, телефонный номер (голосового телефона или факса), либо иной указатель ресурса, воспринимаемый телекоммуникационным оборудованием, адрес электронной почты или другой ресурс местонахождения, который может быть описан унифицированным указателем ресурса (URL).

Адрес задается как URL, дополненный спецификациями времени и кодами использования, помогающими определить, какой адрес следует использовать в данное время и для данной цели.

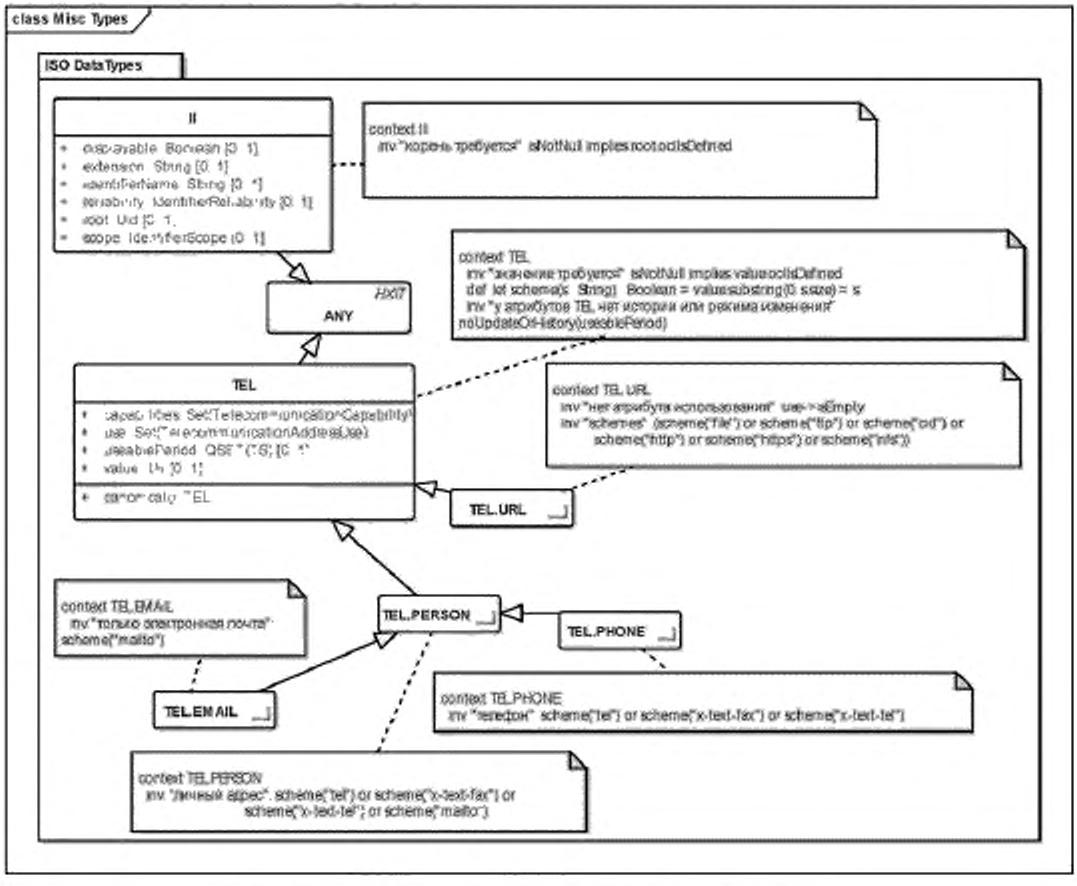


Рисунок 5 — Типы данных идентификации и местонахождения

Значения атрибута value в этом типе данных ограничены унифицированными указателями ресурсов, описанными в документах RFC 1738 и RFC 3966 организации IETF.

Примечание — Этот тип данных призван служить указателем, а не идентификатором ресурса; он используется для указания ресурса местонахождения в форме такого указателя URL, знание которого позволяет

определить местонахождение объекта. Однако в некоторых сценариях для указания ресурса местонахождения используется идентификатор URI. Хотя тип данных TEL позволяет использовать идентификаторы URI, они всегда должны указывать на ресурсы местонахождения. Типичным примером служит использование идентификаторов URI для указания вложений в сообщения SOAP.

7.6.2.2 Синтаксис ИСО/МЭК 11404

```
type TEL = class (
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    value : characterstring,
    use : Set(TelecommunicationAddressUse),
    capabilities : Set(TelecommunicationCapability),
    useablePeriod : QSET(TS)
)
```

7.6.2.3 Атрибуты

7.6.2.3.1 value : Uri: унифицированный идентификатор ресурсов (URI), определенный в соответствии с документом RFC 2396 организации IETF.

Идентификатор URI указывает протокол и контактную точку, определенную этим протоколом для ресурса.

Пример — Тип данных телекоммуникационного адреса широко применяется для представления номеров телефонов и факсов, адресов электронной почты, гиперссылок, ссылок на службы FTP и т. д.

Если атрибут value имеет тип пустоты nullFlavor, то он не обязан содержать правильный указатель URL. К примеру, если причина пустоты имеет значение UNK (не известен), то атрибут value может иметь просто значение «tel:», указывающее, что не известен телефонный номер.

7.6.2.3.2 use : Set(TelecommunicationAddressUse): один или несколько кодов, указывающих системе или пользователю, какой из похожих телекоммуникационных адресов следует использовать при определенной потребности во взаимодействии.

Код использования телекоммуникаций не является полной классификацией типов оборудования или местонахождений. Его основная цель — предложение использования конкретного телекоммуникационного адреса или предостережение от его применения. Не так-то просто определить точные правила, управляющие выбором телекоммуникационного адреса. В объявлениях соответствия может быть уточнено, какие правила могут применять или как именно применяют дополнительные правила.

Непустые значения, присваиваемые атрибуту use, должны браться из системы кодирования HL7 TelecommunicationAddressUse. Текущие значения приведены в таблице 12.

Таблица 12 — Перечисление TelecommunicationAddressUse. ОИД: 2.16.840.1.113883.5.1011

Уровень	Код	Описание	Определение
1	H	home address (домашний адрес)	Домашний телекоммуникационный адрес; попытка контакта по этому адресу для деловых целей может явиться нарушением неприкосновенности личной жизни, и вместо нужного лица может ответить член семьи или иной житель. Обычно используют в экстренных случаях или при отсутсвии контакта
2	HP	primary home (основной домашний адрес)	Основной домашний телекоммуникационный адрес. Используют для контакта с лицом во внерабочее время
2	HV	vacation home (домашний на время отпуска)	Домашний телекоммуникационный адрес на время отпуска. Используют для контакта с лицом во время его отпуска

Окончание таблицы 12

Уровень	Код	Описание	Определение
1	WP	work place (служебный)	Служебный адрес. Первый, по которому должны начинаться попытки контакта в рабочее время
2	DIR	direct (прямой)	Адрес рабочего места или телекоммуникационный адрес, по которому можно связаться с лицом без посредников. У телефона такой номер часто называется «прямым»
2	PUB	public (публичный)	«Стандартный» адрес рабочего места или телекоммуникационный адрес, по которому можно связаться со справочной, общим почтовым ящиком или иным посредником, обеспечивающим контакт с нужным лицом
1	BAD	bad address (плохой адрес)	Признак «плохого», бесполезного адреса
1	TMP	temporary address (временный адрес)	Временный адрес, может быть приемлемым для визита или корреспонденции. История смены адресов может представить более детальную информацию
1	AS	answering service (автоответчик)	Автоответчик, используемый для менее срочных контактов, а также в ситуация, когда основная цель контакта — оставить сообщение или прослушать автоматическое объявление
1	EC	emergency contact (экстренный контакт)	Контакт, специально указанный для связи в экстренных случаях. Первый, по которому должны начинаться попытки контакта в экстренных случаях вне зависимости от наличия адресов с другими кодами использования
1	MC	mobile contact (мобильный контакт)	Мобильное телекоммуникационное устройство, которое владелец носит с собой. Может иметь характеристики других кодов использования, пригоден для срочных контактов, но не является первым, по которому должны начинаться попытки обычных деловых контактов
1	PG	pager (пейджер)	Пейджер, позволяющий инициировать обратный вызов или оставить очень короткое сообщение

Синтаксис ИСО/МЭК 11404 для типа данных атрибута use:

```
type TelecommunicationAddressUse = enumeration (H, HP, HV, WP, DIR,
    PUB, BAD, TMP, AS, EC, MC, PG)
```

7.6.2.3.3 capabilities : Set(TelecommunicationCapability): один или несколько кодов, указывающих системе или пользователю, какие возможности взаимодействия доступны по данному телекоммуникационному адресу.

Непустые значения, присваиваемые атрибуту use, должны браться из системы кодирования HL7 TelecommunicationCapability. Текущие значения приведены в таблице 13.

Таблица 13 — Перечисление TelecommunicationCapability, ОИД: 2.16.840.1.113883.5.1118

Уровень	Код	Описание	Определение
1	voice	Voice (голос)	Это устройство позволяет речевое взаимодействие (например, разговор с другим лицом, запись речи или речевое управление компьютером)
1	fax	Fax (факс)	Это устройство может получать факсимильные сообщения
1	data	Data (данные)	Это устройство может получать данные (например, модем)
1	tty	Text (текст)	Это устройство является телетайпом

Окончание таблицы 13

Уровень	Код	Описание	Определение
1	sms	SMS	Это устройство может получать СМС-сообщения

Синтаксис ИСО/МЭК 11404 для типа данных атрибута capabilities:

```
type TelecommunicationCapability = enumeration (voice, fax, data,
tty, sms)
```

7.6.2.3.4 useablePeriod : QSET(TS): периоды времени, в течение которых можно пользоваться данным телекоммуникационным адресом.

Для телефонного номера это могут быть периоды времени, в течение которого нужному абоненту можно звонить по этому номеру. Для веб-адреса это могут быть периоды времени, в течение которого содержание сайта предполагается доступным по этому адресу.

7.6.2.4 Равенство

Два непустых значения типа TEL равны, если их канонические формы имеют одно и то же значение атрибутов value. Атрибуты use и useablePeriod исключаются из проверки на равенство.

7.6.2.5 Инварианты:

- атрибут value должен присутствовать.

Представление инвариантов на языке OCL:

```
inv "атрибут value требуется": isNotNull implies value.oclIsDefined
def: let scheme(s : String) : Boolean
    = value.substring(0, s.size) = s
inv "у атрибутов типа данных TEL нет истории или режима обновления":
    noUpdateOrHistory(useablePeriod)
```

7.6.2.6 Операции

7.6.2.6.1 canonical : TEL: форма телекоммуникационного адреса, из которого удалены все разделители и другие незначащие символы.

Синтаксис схемы tel: позволяет указывать в номере телефона такие символы, как «(» и «)», представляющие собой синтаксические разделители, не меняющие собственно телефонный номер. Функция canonical вырезает эти символы из адресной части URL. Какие именно символы вырезаются, зависит от схемы.

Синтаксис схемы mailto: позволяет указывать дополнительное имя и данные заголовка. Эта дополнительная информация вырезается из канонической формы схемы mailto:, оставляя только чистый(е) адрес(а) электронной почты.

7.6.2.7 Расширения синтаксиса URL/URI syntax

В настоящем стандарте определены следующие расширения схемы URL:

- x-text-tel: — указывает, что принимающим устройством является телетайп; синтаксис адресной части URL тот же, что у TEL;

- x-text-fax: — указывает, что принимающим устройством является факсимильный аппарат; синтаксис адресной части URL тот же, что у TEL. Этот протокол заменяет протокол fax, запрещенный организацией W3C.

В настоящем стандарте определены следующие расширения схемы URN:

– hl7ii — ссылка на значение типа II, определенное в настоящем стандарте. Полный синтаксис этого расширения URN имеет следующий вид: urn:hl7ii:{root}[:{extension}], где {root} {extension} (если присутствует) являются компонентами значения типа II, на которое дается ссылка. Полные детали этого протокола определены в спецификации HL7 Abstract Data Types.

7.6.2.8 Примеры

7.6.2.8.1 Веб-адрес

```
<example xsi:type="TEL" value="http://www.temp.org/example/234232"/>
```

Ссылка на веб-страницу, доступную по адресу gost.e-ecolog.ru

7.6.2.8.2 Телефон, являющийся одновременно домашним и служебным

```
<example xsi:type="TEL" value="tel:+15556755745"
    use="H WP" capabilities="voice fax"/>
```

Номер домашнего телефона (H) лица, которое работает на дому (WP). Телефон способен принимать голосовые и факсимильные вызовы.

7.6.2.8.3 Неизвестный номер домашнего телефона

```
<example xsi:type="TEL" nullFlavor="UNK" value="tel:" use="H"/>
```

Неизвестный номер домашнего телефона (H).

7.6.2.8.4 Рабочий телефон с дополнительным номером

```
<example xsi:type="TEL" value="tel:+1(555)6755745;postd=545" use="WP"/>
```

Указан рабочий телефон с дополнительным номером. Дополнительные номера используются не только в последовательности, набираемой после основного номера. Дополнительные детали приведены в документе RFC 3966 организации IETF. Каноническая форма для этого примера имеет вид <tel va lue="tel:+15556755745;postd=545" use="WP"/>.

7.6.3 Тонкость TEL.URL

7.6.3.1 Описание

Ограничение типа данных TEL

Тонкость TEL.URL ограничивает тип данных TEL таким образом, что он должен указывать на ресурс местонахождения, возвращающий двоичное содержание.

7.6.3.2 Инварианты

- код использования отсутствует;
- схема URL должна быть file, nfs, ftp, cid (для вложений в SOAP), http или https.

Представление инвариантов на языке OCL:

```
inv "код использования отсутствует": use->isEmpty
inv "схемы": (scheme("file") or scheme("ftp") or
    scheme("cid") or scheme("http") or scheme("https")
    or scheme("nfs"))
```

7.6.4 Тонкость TEL.PERSON

7.6.4.1 Описание

Ограничение типа данных TEL

Тонкость TEL.PERSON ограничивает тип данных TEL таким образом, что он должен указывать метод коммуникации с физическим лицом.

7.6.4.2 Инварианты:

- схема URL должна быть tel, x-text-fax, x-text-tel или mailto.

Представление инвариантов на языке OCL:

```
inv "телекоммуникационный адрес физического лица": scheme("tel") or
    scheme("x text fax")or scheme("x-text tel") or
    scheme("mailto")
```

7.6.5 Тонкость TEL.PHONE

7.6.5.1 Описание

Ограничение типа данных TEL

Тонкость TEL.PERSON ограничивает тип данных TEL таким образом, что он должен указывать некоторую систему коммуникации с физическим лицом, основанную на телефонии.

7.6.5.2 Инварианты

- схема URL должна быть tel, x-text-fax или x-text-tel.

Представление инвариантов на языке OCL:

```
inv "телефон": scheme("tel") or scheme("x-text-fax") or
    scheme("x text tel")
```

7.6.6 Тонкость TEL.EMAIL

7.6.6.1 Описание

Ограничение типа данных TEL

Тонкость TEL.EMAIL ограничивает тип данных TEL адресом электронной почты.

7.6.6.2 Инварианты

- схема URL должна быть mailto.

Представление инвариантов на языке OCL:

```
inv "только электронная почта": scheme("mailto")
```

7.6.7 Тип данных II (идентификатор экземпляра)

7.6.7.1 Описание

Специализация типа данных ANY

Уникальный идентификатор предмета или объекта.

Пример — Объектный идентификатор объектов, определенных в эталонной информационной модели HL7 RIM, номер медицинской карты, идентификатор направления, идентификатор услуги в прейскуранте, регистрационный номер транспортного средства и т. д.

Примечание — Идентификаторы экземпляров обычно определены на основе объектных идентификаторов ИСО.

С помощью идентификатора можно выбрать одну запись, один объект или предмет из множества кандидатов. Обычно идентификатор сам по себе, без контекста, бесполезен. Идентификаторы отличаются от дескрипторов понятий тем, что последние никогда не идентифицируют отдельный предмет, хотя иногда понятие может быть представлено отдельной записью или объектом.

Элементы обработки информации, объявляющие непосредственное или косвенное соответствие настоящему стандарту, никогда не должны предполагать, что приложения-получатели могут вывести из идентификатора или его компонентов информацию об органе, присвоившем идентификатор, или о типе идентификатора.

7.6.7.2 Синтаксис ИСО/МЭК 11404

```
type II = class {
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    root : characterstring,
    extension : characterstring,
    identifierName : characterstring,
    displayable : boolean,
    scope: IdentifierScope,
    reliability : IdentifierReliability
    )
```

7.6.7.3 Атрибуты

7.6.7.3.1 root : Uid: уникальный идентификатор, гарантирующий глобальную уникальность идентификации объекта.

Если атрибуту root присвоено значение и нет ни причины пустоты nullFlavor, ни атрибута extension, то значение атрибута root само по себе является глобально уникальным идентификатором. В присутствии

непустого атрибута extension значение атрибута root является уникальным идентификатором «пространства имен», из которого берутся значения идентификатора, содержащегося в атрибуте extension. Значение атрибута root вовсе не обязательно коррелирует с организацией, присваивающей идентификаторы. Конкретная организация может управлять несколькими пространствами имен идентификаторов, а контроль над конкретным пространством имен может с течением времени переходить от одной организации к другой, в то время как значение атрибута root будет продолжать оставаться неизменным.

Это поле может представлять собой универсально уникальный идентификатор DCE UUID, объектный идентификатор (ОИД) или специальный идентификатор из списка, публикуемого организацией ИСО или рабочей группой HL7.

Сравнение значений атрибута root всегда чувствительно к регистру. Идентификаторы UUID должны быть представлены в верхнем регистре, так что за этим всегда надо следить.

Атрибут root не должен иметь какое-либо смысловое значение; его назначение заключается только в обеспечении глобальной вычислительной уникальности.

7.6.7.3.2 extension : String: строка символов, представляющая собой уникальный идентификатор в пространстве имен, определенном значением атрибута root.

Схема, состоящая из атрибутов root и extension, означает, что конкатенация значений этих атрибутов должна быть глобально уникальным идентификатором элемента, идентифицируемого данным значением типа II.

В некоторых схемах идентификации определены определенные стили представления значений идентификаторов. Например, в США номер карточки социального страхования SSN (Social Security Number) обычно записывается с дефисом по шаблону «123-12-1234». Однако дефисы не являются значащими и такой номер SSN может быть представлен в форме «123121234» без дефисов. В трм случае, если схемы идентификации предусматривают несколько вариантов представления идентификаторов, рабочая группа HL7 или организация ИСО могут принять правило выбора предпочтительной формы и указать его в документе, описывающем соответствующую внешнюю схему идентификации.

Если атрибут extension отсутствует в непустом значении типа II, то значение атрибута root представляет полный уникальный идентификатор. Если же значение атрибута root не является полным уникальным идентификатором, а значение атрибута extension неизвестно, то значение типа II должно иметь причину пустоты nullFlavor даже в том случае, если атрибут root имеет непустое значение.

7.6.7.3.3 identifierName : String: человеко-итаемое имя пространства имен, представленного атрибутом root.

Примечание — Это описательное имя фактического пространства имен, например, «California, US Driving Licence Number, 1970-». Значение атрибута identifierName не отсылает к организации, присвоившей идентификатор (например, California Department of Motor Vehicles). Оно призвано служить в качестве человекочитаемой метки, если идентификатор должен быть представлен человеку, для которого ОИД не является значащим.

Имя идентификатора не имеет вычислительного характера. Значение атрибута identifierName никогда не может изменить смысл атрибута root. Имя идентификатора должно без посторонней помощи обеспечить человеку, интерпретирующему значение типа II, информацию об организации, присвоившей идентификатор. Прикладные программы не должны пытаться выполнить какое-либо принятие решений, определить совпадение, выполнить фильтрацию или иную обработку, основанную на присутствии этого свойства или его значении. Оно предназначено только для отображения и помощи разработчикам.

Вся логика принятия решений должна быть основана исключительно на значениях атрибутов root и extension. В элементах обработки информации, объявляющих непосредственное или косвенное соответствие настоящему стандарту, атрибут identifierName может быть не реализован, но они не должны отклонять экземпляры, в которых этот атрибут присутствует.

Вообще говоря, атрибут identifierName рекомендуется указывать только в том случае, если присутствует атрибут extension, что позволяет, например, вывести на экран строку «California, US Driving Licence Number, 1970-: 123456789». Кроме того, что этот текст является исключительно описателем пространства имен, никакие другие указания по его содержанию не даются, например, «Driving Licence» или даже «California Driving Licence» не были бы идеальными значениями этого атрибута. Однако форматирование, преобразование в верхний регистр, пробельные элементы, язык и т. д. полностью оставляются на усмотрение отправителя.

7.6.7.3.4 displayable : Boolean: если идентификатор предназначен для представления человеку и ввода данных, то этот атрибут должен иметь значение true, если только для машинной обработки — значение false.

В элементах обработки информации, объявляющих непосредственное или косвенное соответствие настоящему стандарту, атрибут displayable может быть не реализован, но они не должны отклонять экземпляры, в которых этот атрибут присутствует.

7.6.7.3.5 scope : IdentifierScope: область, в которой идентификатор применяется к ассоциированному с ним объекту.

Непустые значения, присваиваемые атрибуту scope, должны браться из системы кодирования HL7 IdentifierScope. Текущие значения приведены в таблице 14.

Таблица 14 — Перечисление IdentifierScope, ОИД: еще не присвоен

Уровень	Код	Описание	Определение
1	BUSN	Деловой идентификатор	Идентификатор, чья область определена деловой практикой, ассоциированной с объектом. В отличие от других областей, деловая практика может позволять повторно использовать идентификатор данного объекта для других, тесно связанных объектов
1	OBJ	Идентификатор объекта	Идентификатор, присвоенный конкретному объекту. Он остается тем же, даже когда объект претерпевает переходы состояний
1	VER	Идентификатор версии	Идентификатор, относящийся к состоянию конкретного объекта в данный момент времени. Идентификатор будет меняться при каждом переходе состояния объекта, например, идентификатор объекта до перехода в состояние «приостановки» отличается от идентификатора объекта после перехода состояния. Каждый идентификатор версии может быть связан только с одним событием управляющего действия (ControlAct), которое привело к возникновению данной версии (хотя управляющее действие может никогда не иметь экземпляров). Приложения, не поддерживающие версионность объектов, должны игнорировать и не сохранять эти идентификаторы во избежание недоразумений, связанных с оставлением того же самого идентификатора версии объекту, претерпевшему изменения
1	VW	Идентификатор, специфичный для представления	Идентификатор конкретного моментального снимка версии объекта. Он идентифицирует представление делового объекта в конкретный момент времени и как таковой идентифицирует множество элементов данных, которые могут быть аттестованы или заверены электронной подписью. Этим он отличается от идентификатора версии, который в заданный момент времени идентифицирует не объект, но объем информации об этом объекте. Этот идентификатор следует изменять при каждом преобразовании информации (например, при добавлении переводов кода или упрощенного текстового представления, а также при добавлении информации об объекте, описывающей его в конкретный момент времени)

Синтаксис ИСО/МЭК 11404 для атрибута identifierScope:

```
type IdentifierScope = enumeration (BUSN, OBJ, VER, VW)
```

7.6.7.3.6 reliability : IdentifierReliability: известная надежность идентификатора. Значение этого атрибута может использоваться в алгоритмах сопоставления идентификаторов.

Непустые значения, присваиваемые атрибуту reliability, должны браться из системы кодирования HL7 IdentifierReliability. Текущие значения приведены в таблице 15.

Таблица 15 — Перечисление IdentifierReliability, ОИД: еще не присвоен

Уровень	Код	Описание	Определение
1	ISS	Issued by system (присвоен системой)	Идентификатор присвоен системой, ответственной за создание экземпляра типа II

Окончание таблицы 15

Уровень	Код	Описание	Определение
1	VRF	Verified by system (проверен системой)	Идентификатор не был присвоен системой, ответственной за создание экземпляра типа II, однако эта система проверила правильность идентификатора у организации, присвоившей идентификатор, или у другой системы, проверившей этот идентификатор
1	UNV	Unverified by system (не проверен системой)	Идентификатор был предоставлен системе, ответственной за создание экземпляра типа II, но не был проверен, например, номер водительских прав был введен в систему пользователем вручную

Синтаксис ИСО/МЭК 11404 для атрибута reliability:

```
type IdentifierReliability = enumeration (ISS, VRF, UNV)
```

7.6.7.4 Равенство

Два идентификатора экземпляра равны только в том случае, если у них нет причины пустоты nullFlavor, значения атрибутов root равны, а значения атрибутов extension либо оба пусты, либо равны. Атрибуты displayable, identifierName, scope и reliability игнорируются, хотя свойства scope и reliability могут быть использованы для определения того, насколько равенство значимо в данном контексте.

7.6.7.5 Инварианты

- если значение типа II не имеет причины пустоты nullFlavor, атрибут root должен присутствовать.

Представление инвариантов на языке OCL:

```
inv "атрибут root требуется": isNotNull implies root.oclIsDefined
```

7.6.7.6 Комментарии к технической спецификации ИСО/ТС 22220

В технической спецификации ИСО/ТС 22220 определены четыре поля идентификаторов субъекта медицинской помощи: обозначение, географическая область, источник и тип. Из них только первый, обозначение, относится к области применения типа данных II. Обозначение определено как число или код, присваиваемый лицу организацией, учреждением, агентством или местными властями в целях уникальной идентификации субъекта здравоохранения в пределах этой организации здравоохранения, учреждения, агентства или территории.

Тип данных II выполняет эту роль, обеспечивая предоставление уникального идентификатора. Когда тип данных II используется для идентификации субъекта медицинской помощи, то контекст его использования должен обеспечить поддержку свойств географической области, источника и типа.

Примечание — В типе данных II предусмотрено поле имени идентификатора (identifierName), но оно не обеспечивает формальную поддержку идентификации источника идентификатора.

7.6.7.7 Примеры

7.6.7.7.1 Водительские права

```
<example xsi:type="II" root="2.16.840.1.113883.12.333"
    extension="45634353344"
    reliability="UNV" scope="BUSN"/>
```

ОИД 2.16.840.1.113883.12.333 был создан рабочей группой HL7 как общий идентификатор уполномоченных организаций по выдаче водительских прав. Конкретный номер водительских прав содержится в атрибуте extension. Атрибут надежности имеет значение UNV — номер водительских прав был введен в систему, но она не может его проверить. Атрибут scope для водительских прав имеет значение BUSN, поскольку номер водительских прав может использоваться для идентификации нескольких разных объектов, ассоциированных с одним и тем же пациентом. В примере использован общий ОИД уполномоченных организаций по выдаче водительских прав, но его не рекомендуется использовать на практике, поскольку таких организаций много и выданные ими номера могут пересекаться.

7.6.7.7.2 Номер карточки социального страхования SSN в США

```
<example xsi:type="II" root="2.16.840.1.113883.4.1" extension="123456789"
    reliability="UNV" scope="BUSN"/>
```

Хотя этот идентификатор часто форматируют по шаблону 123-45-6789, дефисы «-» следует удалить, поэтому правильный формат — 123456789.

7.6.7.7.3 Номер пациента, присвоенный организацией NHS

```
<example xsi:type="II" root="2.16.840.1.113883.2.1.4.1"
    extension="9999999484"
    reliability="VRF" scope="BUSN"/>
```

Это пример номера пациента, присваиваемого организацией NHS в Англии и Уэльсе. ОИД для атрибута root, идентифицирующий организацию NHS, равен 2.16.840.1.113883.2.1.4.1.

Поскольку номер, присвоенный NHS, многократно используется в разных экземплярах клинических документов, относящихся к одному и тому же пациенту, а также во многих других записях, то обычно у него атрибут scope не равен OBJ, за исключением ситуации, когда этот номер используется в главном регистре NHS.

Как правило, атрибут scope имеет значение BUSN. В большинстве информационных систем учреждений, управляемых организацией NHS, предусмотрены программные интерфейсы к главному регистру пациентов, который ведет эта организация, или к другим системам, взаимодействующим с этим регистром, поэтому в этих случаях атрибут надежности reliability имеет значение «VRF» (проверен системой).

7.6.7.7.4 Номер пациента в австралийской системе medicare

```
<example xsi:type="II" root="1.2.36.174030967" extension="1234567892"
    reliability="VRF" scope="OBJ"/>
```

Это пример номера пациента в австралийской системе medicare. Значением атрибута root служит корневой ОИД организации HIC, присваивающей такие номера. (Следовало бы указать более детальный ОИД, но в настоящее время эта организация еще не опубликовала схему своей ветви ОИД.) Атрибут надежности reliability имеет значение «VRF», поскольку система-отправитель проверила правильность номера пациента в организации HIC с помощью одного из ее электронных интерфейсов. Атрибут scope имеет значение OBJ, так как этот номер используется только для идентификации лицевого счета семьи пациента, который ведет организация HIC.

7.6.7.7.5 Идентификатор записи

```
<example xsi:type="II" root="D6A7AB37 4220 4D80 9052 8A4959A203E3"
    reliability="ISS" scope="VER"/>
```

Идентификатор UUID, присвоенный системой-отправителем конкретной версии записи, представленной значением типа II.

7.6.7.7.6 Номер результата лабораторного анализа

```
<example xsi:type="II" root="2.16.840.1.113883.19.5.34" extension="2345344"
    reliability="ISS" scope="OBJ"/>
```

Номер, идентифицирующий результат лабораторного анализа. Он имеет атрибут scope со значением «OBJ» и остается неизменным в течение всего процесса лабораторного анализа, включая получение образца, его обработку, подготовку документа с результатом анализа и подпись этого документа. Этот номер сообщается лабораторной информационной системой, присвоившей данный идентификатор.

7.6.7.7.7 Идентификатор компонента медицинской карты, соответствующего стандарту ИСО 13606

```
<example xsi:type="II" root="2.16.840.1.113883.19.5.462" extension="976295765"
```

```
reliability="VRF" scope="VER"/>
```

В этом примере показано значение идентификатора rc_id, определенного в ИСО 13606-1 (идентификатор компонента медицинской карты), остающегося неизменным в различных хранилищах электронных медицинских карт, однако меняющегося для различных версий компонента и не сохраняющегося в рабочем процессе перехода состояний компонента. Поэтому атрибут scope имеет значение «VER» (идентификатор версии).

Значение атрибута надежности reliability равно VRF; в контексте ИСО 13606 использование значения ISS не рекомендовано, поскольку оно может быть использовано только системой, которая его присвоила, а это создает сложности для аттестации содержания компонента.

Значение ISS в основном используется при управлении идентификацией пациентов, поскольку может способствовать процессу объединения/разъединения/слияния идентификаторов пациента.

7.6.7.7.8 Сообщение с моментальным снимком

```
<example xsi:type="DSET_II">
<item root="2.16.840.1.113883.19.5.971" extension="763491"
    reliability="VRF" scope="OBJ"/>
<item root="2.16.840.1.113883.19.5.972" extension="351324"
    reliability="VRF" scope="VER"/>
<item root="0282CA34 2E4E 4B9D 82A5 BD2BF8497940"
    reliability="ISS" scope="VW"/>
</example>
```

В этом примере показано полное использование типа данных DSET(II) в сообщении. В этом сообщении передается моментальный снимок объекта, являющегося экземпляром класса, у которого атрибут идентификатора имеет тип DSET(II). Информация об идентификаторе экземпляра класса, имеющего атрибут DSET(II), основана на объекте с идентификатором 763491 в пространстве имен ОИД 2.16.840.1.113883.19.5.971. Версия объекта, для которого сделан моментальный снимок, имеет идентификатор 351324 в пространстве имен ОИД 2.16.840.1.113883.19.5.972. Наконец, моментальному снимку присвоен свой собственный идентификатор UUID 0282CA34-2E4E-4B9D-82A5-BD2BF8497940, который может использоваться при регистрации данного моментального снимка в системном журнале.

7.7 Типы данных фамилии, имени, отчества и адреса

7.7.1 Введение

Эти типы данных предназначены для передачи фамилий, имен, отчеств и адресов (см. рисунок 6).

7.7.2 Тип данных XP (name or address part)

Абстрактный частный тип

7.7.2.1 Описание

Часть фамилии, имени, отчества или адреса. Каждая часть представляет собой строку символов, которая может быть кодированной, и может также иметь тип причины пустоты nullFlavor. Содержание строк всегда должно быть указано независимо от того, присутствует код или нет.

7.7.2.2 Синтаксис ИСО/МЭК 11404

```
type XP = class {
    nullFlavor : NullFlavor,
    value : characterstring,
    code : characterstring,
    codeSystem : characterstring,
    codeSystemVersion : characterstring,
    language : characterstring,
}
```

7.7.2.3 Атрибуты

7.7.2.3.1 nullFlavor : NullFlavor: если часть не имеет правильного значения, в этом атрибуте должна быть указана причина.

Дальнейшая информация о причине пустоты nullFlavor приведена в 7.3.3.3.1.

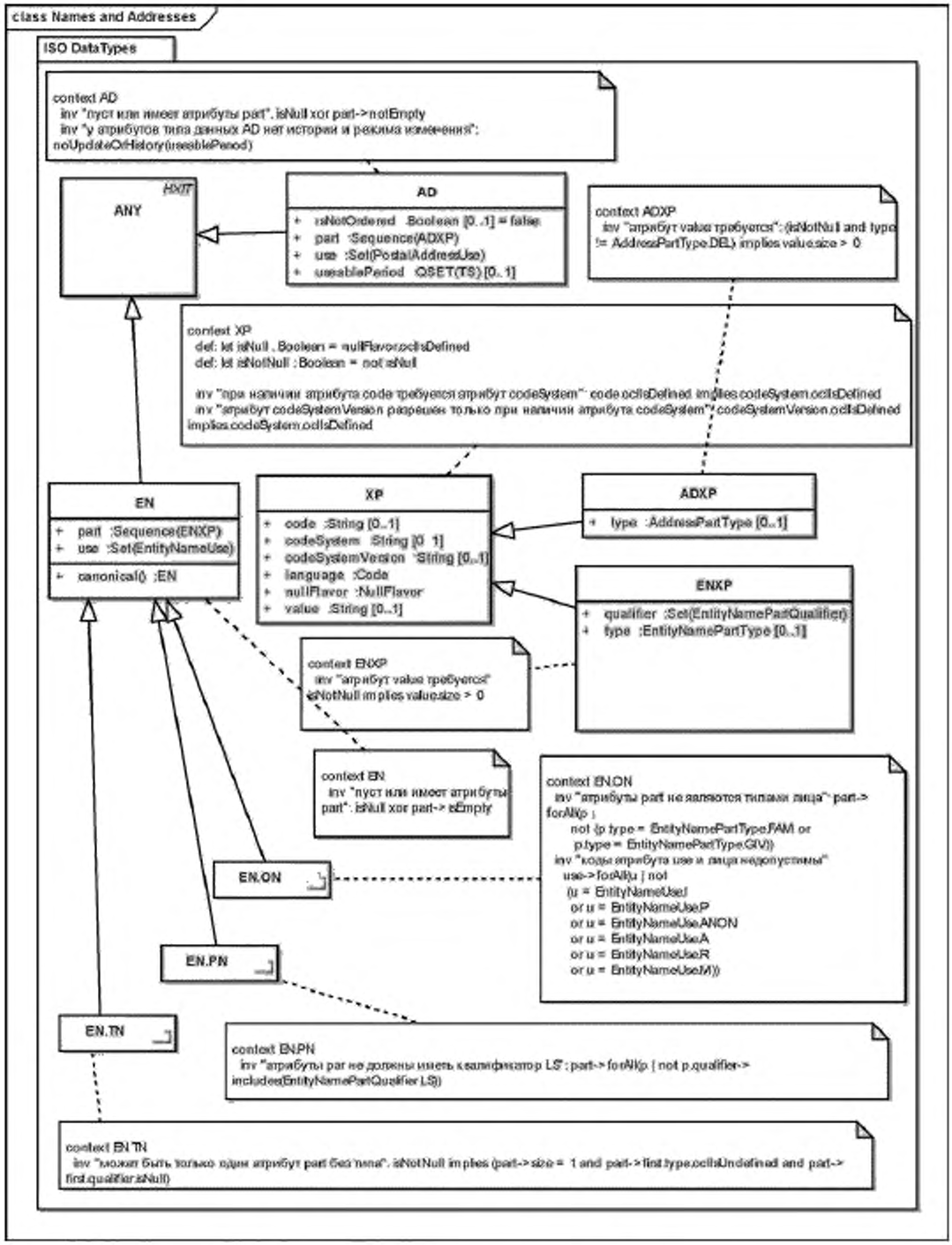


Рисунок 6 — Типы данных фамилии, имени, отчества и адреса

7.7.2.3.2 value: String : фактическое строковое значение части. Если причина пустоты nullFlavor отсутствует, то в этом атрибуте должно присутствовать некоторое содержание.

7.7.2.3.3 code: String : код, присвоенный части в некоторой системе кодированияпри ее наличии.

7.7.2.3.4 codeSystem : String: система кодирования, из которой взято значение атрибута code.

Выбор системы кодирования зависит от типа части, определенного в конкретной специализации. Если атрибут code имеет значение, то должен иметь значение и атрибут codeSystem.

7.7.2.3.5 codeSystemVersion : String: версия системы кодирования, если таковая должна быть указана.

Если атрибут codeSystemVersion имеет значение, то должен иметь значение и атрибут codeSystem.

7.7.2.3.6 language . Code: человеческий язык содержания. Допустимые коды берутся из документа RFC 3066 организации IETF. Если этот атрибут пуст, то язык может быть идентифицирован откуда-то еще, например, из содержания или из тега языка в кодировке unicode.

Хотя для частей может быть указан язык, значение части не зависит от языка, и приложения не должны требовать указывать лингвистическое происхождение любой части фамилии, имени, отчества или адреса.

7.7.2.4 Равенство

Для значений типа XP понятие равенства не определено.

7.7.2.5 Инварианты:

– если атрибут code имеет значение, то атрибут codeSystem должен иметь значение;

– атрибут codeSystemVersion может иметь значение только в случае, если атрибут codeSystem имеет значение.

Определение инвариантов на языке OCL:

```
def: let isNull : Boolean = nullFlavor.oclIsDefined
def: let isNotNull : Boolean = not isNull
inv "при наличии атрибута code требуется атрибут codeSystem":
    code.oclIsDefined implies
    codeSystem.oclIsDefined
inv "атрибут codeSystemVersion разрешен только при наличии атрибута
    codeSystem": codeSystemVersion.oclIsDefined implies
    codeSystem.oclIsDefined
```

7.7.3 ADXP (компонент адреса)

7.7.3.1 Описание

Специализация типа данных XP

Компонент может иметь тег типа, обозначающий его роль в адресе. Типичными компонентами почти любого адреса являются название улицы, номер дома или почтового ящика, почтовый индекс, город, страна, но на региональном, национальном или даже на ведомственном уровне (например, для адресов воинских частей) могут быть определены дополнительные роли.

Адреса обычно разбиваются на строки. Признаком перехода на следующую строку могут служить специальные ограничивающие символы (например, DEL).

7.7.3.2 Синтаксис ИСО/МЭК 11404

```
type ADXP = class (
    nullFlavor : NullFlavor,
    value : characterstring,
    code : characterstring,
    codeSystem : characterstring,
    codeSystemVersion : characterstring,
    language : characterstring,
    type : AddressPartType,
)
```

7.7.3.3 Атрибуты

7.7.3.3.1 type : AddressPartType: указывает, означает ли компонент адреса улицу город, страну, почтовый индекс, почтовый ящик и т. д. Если это свойство имеет пустое значение, то компонент адреса является не классифицированным и должен представлять собой неструктурированный адрес.

Непустые значения, присваиваемые атрибуту type, должны браться из системы кодирования HL7 AddressPartType. Текущие значения приведены в таблице 16.

Таблица 16 — Перечисление AddressPartType, ОИД: 2.16.840.1.113883.5.16

Уровень	Код	Описание	Определение
1	AL	Address line (строка адреса)	Строка адреса представляет собой либо дополнительный указатель, либо адрес доставки, либо адрес улицы. Адрес обычно включает либо строку адреса доставки, либо строку адреса улицы, но не обе вместе
2	ADL	Additional locator (дополнительный указатель)	Компонент с таким свойством может обозначать некоторую единицу адреса, например, номер квартиры, комнаты или этажа. В одном адресе может быть несколько таких компонентов (например, «3-й этаж, кв. 342»). Это может быть обозначением места, находящегося в стороне от адреса, а не меньшей единицей внутри некоторой большей (например, в Нидерландах обозначение «t.o.» означает «напротив» и используется для указания жилых лодок, стоящих напротив фасадов домов, выходящих на улицу)
3	UNID	Unit identifier (идентификатор единицы)	Номер или название конкретной части здания или комплекса, присвоенные в этом здании или комплексе
3	UNIT	Unit designator (обозначение единицы)	Указывает тип конкретной части здания или комплекса, например, квартира, этаж
2	DAL	Delivery address line (строка адреса доставки)	Строка адреса доставки часто используется вместо выделения способа доставки, почтового накопителя и т. д. Обычно в адресе указывают либо строку с адресом улицы, либо строку с адресом доставки, но не обе вместе
3	DINST	Delivery installation type (тип почтового накопителя)	Указывает тип почтового накопителя (места, в которое письмо должно быть доставлено перед окончательной передачей с помощью способа доставки). Примеры: почтовое отделение, почтовый узел, центральный телеграф, станция
3	DINSTA	Delivery installation area (территория почтового накопителя)	Местонахождение почтового накопителя, обычно поселок или город. Требуется только в тех случаях, когда оно отличается от муниципального образования. Территория, на которой осуществляется доставка почты этим накопителем или службой, например, доставка почтальоном, доставка по сельскому или городскому маршруту
3	DINSTQ	Delivery installation qualifier (квалификатор почтового накопителя)	Номер, буква или имя, выделяющие почтовый накопитель. Например, для «Станции А» квалификатором, выделяющим этот накопитель, будет буква «А»
3	DMOD	Delivery mode (метод доставки)	Тип предлагаемой услуги, метод доставки. Например, почтовый ящик (в почтовом отделении), сельский маршрут, общая доставка
3	DMODID	Delivery mode identifier (идентификатор метода доставки)	Идентификатор метода доставки, например, номер маршрута доставки. Конкретизирует метод доставки (например, указывает номер почтового ящика или сельского маршрута)
2	SAL	Street address line (строка адреса улицы)	Адрес улицы часто используется вместо выделения номера здания, названия улицы, типа улицы и т. д. Обычно в адресе указывают либо строку с адресом улицы, либо строку с адресом доставки, но не обе вместе

Продолжение таблицы 16

Уровень	Код	Описание	Определение
3	BNR	Building number (номер здания)	Номер здания, дома или участка, находящегося на улице. Называется также «основным уличным номером». Это скорее номер здания, нежели улицы
4	BNN	Building number numeric (числовой компонент номера здания)	Числовой компонент номера здания
4	BNS	Building number suffix (суффикс номера здания)	Любая буква, дробь или иной текст, который может быть указан после числового компонента номера здания
3	STR	Street name (название улицы)	Название улицы
4	STB	Street name base (базовое название улицы)	Базовое название проезда по улице, присвоенное муниципалитетом (исключая тип и направление улицы)
4	STTYP	Street type (тип улицы)	Тип улицы (например, улица, проспект, площадь и т. д.)
3	DIR	Direction (направление)	Направление улицы [например, N (Северная), S (Южная), W (Западная), E (Восточная)]
2	INT	Intersection (перекресток)	Означает, что фактический адрес находится на перекрестке двух или более улиц или рядом с ним
1	CAR	Care of (через)	Название получателя почтового отправления, который находится по этому адресу и обеспечит передачу отправления его адресату. Примечание — Этот тип включен только для поддержки соглашений о написание адресных строк «c/-». Его нельзя использовать, если информация доверяется одной стороне по поручению другой каким-либо значащим способом
1	CEN	Census tract (переписной район)	Географическая субъединица, выделяемая в демографических целях
1	CNT	Country (страна)	Страна
1	CPA	County or parish (графство или приход)	Административная единица штата или провинции [в 49 штатах США используется термин «county» (графство), в Луизиане — «parish» (приход)]
1	CTY	Municipality (муниципалитет)	Название города, поселка, деревни или иного муниципального образования либо центра доставки
1	DEL	Delimiter (разделитель)	Разделители печатаются без окаймления пробельными символами. Если этот компонент пуст, то разделителем служит переход на следующую строку
1	POB	Post box (почтовый ящик)	Пронумерованный ящик в почтовом отделении
1	PRE	Precinct (избирательный участок)	Структурная единица муниципального образования
1	STA	State or province (штат или провинция)	Административная единица страны, которая в федеральных государствах сохраняет ограниченный суверенитет

Окончание таблицы 16

Уровень	Код	Описание	Определение
1	ZIP	Postal code (почтовый индекс)	Почтовый индекс, обозначающий район, обслуживаемый почтовым отделением
1	DPID	Delivery point identifier (идентификатор места доставки)	Значение, однозначно идентифицирующее почтовый адрес
Примечание — Иерархический характер этой системы кодирования скорее относится к композиции, а не к категоризации, например, «название улицы» является компонентом «строки адреса улицы».			

Синтаксис ИСО/МЭК 11404 атрибута type:

```
type AddressPartType = enumeration (AL, ADL, UNID, UNIT, DAL, DINST,
    DINSTA, DINSTQ, DMOD, DMODID, SAL, BNR, BNN, BNS, STR, STB, STTYP,
    DIR, INT, CAR, CEN, CNT, CPA, CTY, DEL, POB, PRE, STA, ZIP)
```

7.7.3.4 Равенство

Два значения типа ADXP равны, если попарно равны значения их атрибутов type и value. Атрибуты code и language игнорируются.

Примечание — Разъяснение: два пустых значения атрибутов type считаются равными.

7.7.3.5 Инварианты:

- если атрибут type не пуст, то атрибут value может быть пустым только в том случае, когда значение атрибута type равно DEL.

Определение инвариантов на языке OCL:

```
inv "атрибут value требуется": isNotNull implies value.size > 0
```

7.7.3.6 Привязка

В компоненте адреса, у которого атрибут type имеет значение CNT (страна), пара свойств code+codeSystem, унаследованная от типа данных XP, должна быть привязана к двухбуквенным, трехбуквенным или числовым кодам стран, определенным в ИСО 3166. В объявлениях соответствия могут быть указаны привязки других типов компонентов адреса или ограничения выбора кодов стран.

7.7.4 Тип AD (адрес)

7.7.4.1 Описание

Специализация типа данных ANY

Определение: почтовый, домашний или служебный адрес. Тип данных AD прежде всего используется для передачи данных, позволяющих печатать этикетки с адресами или позволяющих лицу физически посетить адресата. Тип данных почтового адреса AD не предназначен служить контейнером для дополнительной информации, которая может быть полезной для поиска географического местонахождения (например, для GPS-координат) или для проведения эпидемиологических исследований. Такая дополнительная информация может передаваться в других, более подходящих структурах данных.

Адреса представляют собой последовательность компонентов, к которым добавлены код использования use и срок действия useablePeriod, указывающие, может ли данный адрес использоваться для конкретной цели и когда может использоваться.

7.7.4.2 Синтаксис ИСО/МЭК 11404

```
type AD = class {
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
```

```
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    part : Sequence(ADXP),
    use : Set(PostalAddressUse),
    useablePeriod : QSET(TS),
    isNotOrdered : boolean
)
```

7.7.4.3 Атрибуты

7.7.4.3.1 part : Sequence(ADXP): последовательность компонентов адреса, например, названия улицы или номера почтового ящика, города, почтового индекса, страны и т. д.

7.7.4.3.2 Свойство use : Set(PostalAddressUse): один или несколько кодов, информирующих систему или пользователя о том, какой из адресов следует выбрать для данной цели.

Адрес, в котором код атрибута use не указан, может быть адресом по умолчанию, пригодным для любых целей, но адрес, в котором этот код указан, является предпочтительным для соответствующей цели.

Непустые значения, присваиваемые атрибуту type, должны браться из системы кодирования HL7 PostalAddressUse. Текущие значения приведены в таблице 17.

Таблица 17 — Перечисление PostalAddressUse, ОИД: 2.16.840.1.113883.5.1012

Уровень	Код	Описание	Определение
1	AddressUse		
2	H	Home address (домашний адрес)	Домашний адрес; попытка контакта по этому адресу для деловых целей может явиться нарушением неприкосновенности личной жизни, и вместо нужного лица по этому адресу может находиться член семьи или иной житель. Обычно используется в экстренных случаях или если других способов контакта нет
3	HP	Primary home (основной домашний адрес)	Основной домашний адрес. Используется для контакта с лицом во внерабочее время
3	HV	Vacation home (домашний на время отпуска)	Домашний адрес на время отпуска. Используется для контакта с лицом во время его отпуска
2	WP	Work place (служебный)	Служебный адрес. Первый, по которому должны начинаться попытки контакта в рабочее время
3	DIR	Direct (прямой)	Адрес рабочего места или телекоммуникационный адрес, по которому можно связаться с лицом без посредников. У телефона такой номер часто называется «прямым»
3	PUB	Public (публичный)	«Стандартный» адрес рабочего места или телекоммуникационный адрес, по которому можно связаться со справочной, общим почтовым ящиком или иным посредником, обеспечивающим контакт с нужным лицом
2	BAD	Bad address (плохой адрес)	Признак «плохого», бесполезного адреса
2	PHYS	Physical visit address (адрес физического визита)	Используемый в основном для посещения этого адреса
2	PST	Postal address (почтовый адрес)	Используется для отправки письма

Окончание таблицы 17

Уровень	Код	описание	Определение
2	TMP	Temporary address (временный адрес)	Временный адрес, может быть приемлемым для визита или корреспонденции. История смены адресов может представить более детальную информацию
1	AddressRepresentationUse. Идентифицирует различные представления адреса. Способ представления может влиять на использование адреса (например, для формальных коммуникаций может требоваться идеографическое представление)		
2	ABC	Alphabetic (алфавитное)	Алфавитное представление имени (romaji в Японии)
2	IDE	Ideographic (идеографическое)	Идеографическое представление имени (kanji в Японии, китайские иероглифы)
2	SYL	Syllabic (силлабическое)	Силлабическая транскрипция имени (kana в Японии, hangul в Корее)
1	SRCH	Search type uses (поисковое именование)	Именование, используемое для поиска или определения совпадения
2	SNDX	Soundex	Свертка произношения адреса в соответствии с алгоритмом SoundEx
2	PHON	Phonetic (фонетическое)	Произношение адреса, воспринимаемое оператором ввода, то есть близкая аппроксимация к произношению адреса, не основанная на фонетическом алгоритме

Синтаксис ИСО/МЭК 11404 для атрибута use:

```
type PostalAddressUse = enumeration (H, HP, HV, WP, DIR, PUB, BAD,
    TMP, ABC, IDE, SYL, PHYS, PST, SRCH, SNDX, PHON)
```

7.7.4.3.3 useablePeriod : QSET(TS): общая спецификация времени GTS (General Timing Specification), задающая периоды времени, в течение которых можно пользоваться данным адресом. Используется для указания различных адресов для различных дней недели или года.

7.7.4.3.4 isNotOrdered : Boolean: булевское значение, указывающее, известен порядок компонентов адреса или нет. Хотя компоненты адреса всегда передаются в определенном порядке, тем не менее порядок, в котором они должны быть представлены пользователю, может быть, а может и не быть важным или значимым. Это можно указать с помощью свойства isNotOrdered. По умолчанию этот атрибут имеет значение false.

7.7.4.4 Равенство

Два адресных значения считаются равными, если они имеют одинаковые компоненты адреса вне зависимости от их упорядочения.

Атрибуты use, useablePeriod и isNotOrdered исключаются из проверки на равенство.

Примечания

1 Даже если атрибут isNotOrdered имеет значение false, то есть известно, что порядок представления компонентов адреса важен, при проверке на равенство его не учитывают.

2 Два значения, описывающие тот же самый адрес, но представленные с помощью разных наборов компонентов адреса (возможно, из-за разной детализации структуры), не должны считаться равными.

7.7.4.5 Инварианты:

- значение типа AD или имеет причину пустоты nullFlavor, или по меньшей мере один компонент.

Представление инвариантов на языке OCL:

```
inv "пуст или имеет компоненты": isNull xor part->notEmpty
inv "атрибуты типа данных AD не имеют истории или режима изменения":
    noUpdateOrHistory(useablePeriod)
```

7.7.4.6 Комментарии к технической спецификации ИСО/ТС 22220

Различные компоненты адреса, определенные в технической спецификации ИСО/ТС 22220, отображаются на типы компонентов адреса, и тип адреса отображаются на атрибут use. Признаки точности дат начала и завершения частично отображаются на точность предоставляемых дат.

7.7.4.7 Примеры

7.7.4.7.1 Адрес с расположением компонентов

```
<example xsi:type="AD" use="WP">
    <part value="1050 W Wishard Blvd" />
    <part type="DEL"/>
    <part value="RG 5th floor"/>
    <part type="DEL"/>
    <part value="Indianapolis, IN 46240"/>
</example>
```

Этот служебный адрес состоит из трех неизвестных компонентов и двух разделителей строк. Семантическая значимость ни одному компоненту не приписана.

7.7.4.7.2 Адрес с типами компонентов

```
<example xsi:type="AD" use="WP">
    <part type="AL" value="1050 W Wishard Blvd"/>
    <part type="AL" value="RG 5th floor"/>
    <part type="CTY" value="Indianapolis"/>
    <part type="STA" value="IN"/>
    <part type="ZIP" value="46240"/>
</example>
```

Это тот же самый адрес, но акцент сделан на стандартной распечатке адреса, а не на его представлении. Вероятно, это наиболее распространенная форма представления адреса — несколько строк адреса, после которых указаны город, штат, почтовый индекс и, возможно, страна.

Примечание — Хотя в этом представлении адреса предполагается, что после первых двух строк адреса следуют дополнительные строки, это не вытекает из данного примера (см. 7.7.4.8).

7.7.4.7.3 Типы строк

```
<example xsi:type="AD" use="WP">
    <part type="SAL" value="1050 W Wishard Blvd"/>
    <part type="ADL" value="RG 5th floor"/>
    <part type="CTY" value="Indianapolis"/>
    <part type="STA" value="IN"/>
    <part type="ZIP" value="46240"/>
</example>
```

Это тот же самый адрес, но предназначенный для системы, различающей разные типы строк.

7.7.4.7.4 Полностью типизированные адреса

```
<example xsi:type="AD" use="WP">
    <part type="BNR" value="1050"/>
    <part type="DIR" value="W"/>
    <part type="STB" value="Wishard"/>
    <part type="STTYP" value="Blvd"/>
    <part type="ADL" value="RG 5th floor"/>
    <part type="CTY" value="Indianapolis"/>
    <part type="STA" value="IN"/>
    <part type="ZIP" value="46240"/>
</example>
```

Тот же самый адрес, в котором выделен каждый компонент; такую форму не используют в США. Однако она полезна в Германии, где многие системы хранят номер дома в отдельном поле.

```
<example xsi:type="AD" use="HP">
    <part type="STR" value="Windsteiner Weg"/>
    <part type="BNR" value="54a"/>
    <part type="CNT" code="DEU" codeSystem=" 1.0.3166.1.2"
      value="D"/>
    <part type="ZIP" value="14165"/>
    <part type="CTY" value="Berlin"/>
</example>
```

Это домашний адрес в стандартном формате, принятом в Германии. Для обеспечения интероперабельности страна закодирована в соответствии с ИСО 3166.

7.7.4.7.5 Неизвестные адреса

```
<example xsi:type="AD" use="WP" nullFlavor="UNK"/>
```

Служебный адрес неизвестен.

7.7.4.8 Представление адресов

Основной целью адреса является его представление в поле конверта «Кому». Полностью указанный адрес (содержащий заданные строки) может быть представлен как сочетание текстов различных компонентов, разделенных пробельными символами и явно указанными разрывами строк. Если эти элементы перемещаются в пространство имен xhtml, то содержание типа данных AD может трактоваться непосредственно в формате html.

Поэтому текст адреса всегда должен генерироваться с соответствующими разрывами строк, включенными в адрес. Это позволяет приложениям, не интерпретирующим семантику адреса, воспроизводить его правильно.

Но поскольку единственной модели представления адресов не существует, приложения могут игнорировать разрывы строк, явно указанные в адресах; эти приложения не обязаны следовать представлению, указанному в конкретном адресе.

7.7.5 Тип данных ENXP (часть именования сущности)

7.7.5.1 Описание

Специализация типа данных XP

Компонент именования сущности (фамилии, имени, отчества лица), который может иметь свойство type, обозначающее роль этого компонента в полном именовании, и квалификатор qualifier, детализирующий эту роль (типичными компонентами именований лиц являются имена, фамилии, обращения и т. д.).

7.7.5.2 Синтаксис ИСО/МЭК 11404

```
type ENXP = class {
    nullFlavor : NullFlavor,
    value : characterstring,
    code : characterstring,
    codeSystem : characterstring,
    codeSystemVersion : characterstring,
    language : characterstring,
    type : EntityNamePartType,
    qualifier : Set{EntityNamePartQualifier}
}
```

7.7.5.3 Атрибуты

7.7.5.3.1 type : EntityNamePartType: указывает, является ли данный компонент именем, фамилией, префиксом, суффиксом и т. д.

Не каждый компонент именования должен иметь код типа type. Если этот код не известен, не применим или просто не определен, то это выражается пустым значением (partType.isNull). Например,

по именованию «Rogan Sulma» нельзя сделать уверенное заключение, что является именем, а что фамилией. Может даже оказаться, что «Rogan» — титул.

Если этому атрибуту присвоено значение, то оно должно быть взято из системы кодирования HL7 EntityNamePartType. Текущие допустимые значения указаны в таблице 18.

Таблица 18 — Перечисление EntityNamePartType, ОИД 2.16.840.1.113883.5.1010

Уровень	Код	Описание	Определение
1	FAM	Family (фамилия)	Имя семьи, то есть имя, связывающее с генеалогией. В некоторых культурах (например, в Эритрее) фамилией сына является первое имя его отца
1	GIV	Given (имя)	Имя или отчество (не следует называть его «первым именем», поскольку в написании именования оно не всегда идет первым)
1	TITLE	Title (обращение)	Часть именования, которая служит как обращение в соответствии с академическим, юридическим, должностным или дворянским статусом и т. д. Примечание — Компоненты обращения включают в себя в том числе компоненты именования, следующие после имени и фамилии, например, квалификацию
1	DEL	Delimiter (разделитель)	Разделитель служит только в качестве литерала, печатаемого в данном представлении именования. Он не имеет неявных ведущих и концевых пробельных символов

Синтаксис ИСО/МЭК 11404 атрибута type:

```
type EntityNamePartType = enumeration (FAM, GIV, TITLE, DEL)
```

Если имя пишется через дефис, например, Mary-Ann, то возникает неоднозначность, надо ли использовать два компонента имени с разделителем, или один компонент имени, в котором указан дефис. Можно использовать следующий эмпирический подход: если каждая часть имени или отчества должна превращаться в инициал, то надо использовать два компонента и разделитель.

7.7.5.3.2 qualifier : Set(EntityNamePartQualifier): этот квалификатор содержит ряд кодов, каждый из которых указывает определенную подкатегорию компонента именования, дополняющую основной тип компонента.

Пример — Можно указать, что имя является прозвищем (код CL), фамилия может быть присвоена при вступлении в брак (код SP) или данной при рождении (код BR).

Если этому атрибуту присвоено значение, то оно должно быть взято из системы кодирования HL7 EntityNamePartQualifier. Текущие допустимые значения указаны в таблице 19.

Таблица 19 — Перечисление EntityNamePartQualifier, ОИД 2.16.840.1.113883.5.1122

Уровень	Код	Описание	Определение
1	LS	Legal status (юридический статус)	Для организаций этот суффикс указывает юридическую форму собственности, например, «Inc.», «Co.», «AG», «GmbH», «B.V.» «S.A.», «Ltd.» и т. д.
1	Виды обращений: дополнительная информация о виде обращения		
2	AC	Academic (ученая степень или звание)	Указывает префикс наподобие «Др.», «M.D.» или «Ph.D.», обозначающий ученую степень или звание
2	NB	Nobility (дворянство)	В Европе и Азии все еще есть люди с дворянскими титулами (аристократы). Германское «von» в большей мере является титулом, нежели приставкой. Другими примерами служат «Барон» или «Его Величество Король...» и т. д. В наши дни используется редко, но в некоторых системах все еще присутствует

Окончание таблицы 19

Уровень	Код	Описание	Определение
2	PR	Professional (профессиональная принадлежность)	Люди, воспитанные культурой Британской империи, нередко используют в качестве суффиксов полномочий аббревиатуры своей профессиональной организации
2	HON	Honorific (почтительное)	Почтительное обращение, например, «The Right Honourable» или «Weledelgeleerde Heer»
1	BR	Birth (урожденное)	Именование лица, данное ему при рождении или усыновлении (удочерении). Примечание — Не используется для временных именований, данных при рождении, например, «Baby of Smith» (ребенок матери Смит), которые являются именованиями с кодом использования TEMP (временное)
1	AD	Acquired (приобретенное)	Приобретенное именование лица. Этот компонент именования может быть приобретен при усыновлении (удочерении) или лицо выбрало его по каким-либо другим причинам. Примечание — Этот тип именования отличается от «другого», «псевдонима», «прозвища» тем, что оно приобретено на формальной основе (например, зарегистрировано как часть юридически признаваемого именования)
2	SP	Spouse (супружеское)	Именование, полученное от партнера в семейных отношениях. Обычно фамилия супруга. Следует учесть, что из существования супружеской фамилии нельзя сделать определенный вывод относительно пола супруга
1	MID	Middle Name (отчество, среднее имя)	Обозначает компонент именования как отчество или среднее имя. В принципе, английское понятие «среднее имя» означает все имена, кроме первого. Этот квалификатор может использоваться, чтобы явно указать, какие из имен считаются вторыми. Квалификатор среднего имени может использоваться и с фамилиями. Такое имеет место в Скандинавии, где соответствует понятиям «mellomnavn»/«mellannamn». Они отражают специфичные правила, указывающие, что имена могут быть взяты как «средние» в различных скандинавских странах
1	CL	Callme (предпочтительное)	Указывает компонент именования, предпочтительный при прямом обращении к лицу
1	IN	Initial (инициал)	Указывает, что компонент именования является инициалом. В инициалы обычно не включается заключительная точка, поскольку это может быть не принято в языках, не использующих латинский алфавит. Инициалы могут содержать более одной буквы, например, инициал «Ph.» может соответствовать имени «Philippe», а «Th.» — имени «Thomas»
1	PFX	Prefix (префикс)	Префикс имеет жесткую связь с компонентом, непосредственно следующим за ним. У префикса нет неявных концевых пробельных символов (хотя он может иметь неявный ведущий пробельный символ)
1	SFX	Suffix (суффикс)	Суффикс имеет жесткую связь с компонентом, непосредственно предшествующим ему. У суффикса нет неявных ведущих пробельных символов (хотя он может иметь неявный концевой пробельный символ)

Синтаксис ИСО/МЭК 11404 атрибута qualifier:

```
type EntityNamePartQualifier = enumeration (LS, AC, NB, PR, HON, BR,
    AD, SP, MID, CL, IN, PFX, SFX)
```

Скандинавские понятия «Mellomnavn/Mellannamn» переводятся как «среднее имя», но не совпадают с английским понятием «middle name», которое попросту означает все имена после первого. Квалификаторы префикса (PFX) и суффикса (SFX) взаимно несовместимы. Не допускается указывать их оба применительно к одному и тому же компоненту. Нет необходимости помечать компонент именования, следующий за префиксом, как суффикс, или наоборот.

Примечание — Инициалы могут содержать более одной буквы, если это требуется лингвистическими нормами соответствующего языка. Сокращения, например, «Др» для «Доктор», не являются инициалами.

7.7.5.4 Равенство

Два значения типа EXNP считаются равными, если они имеют попарно равные атрибуты type и value. Атрибуты code, language и qualifier игнорируются.

Атрибуты use, useablePeriod и isNotOrdered исключаются из проверки на равенство.

Примечание — Разъяснение: два пустых значения атрибутов type считаются равными.

7.7.5.5 Инварианты:

- если атрибут part не пуст, то атрибут value не может быть пустым.

Представление инвариантов на языке OCL:

```
inv "атрибут value требуется": isNotNull implies value.size > 0
```

7.7.5.6 Привязка

В объявлениях соответствия могут быть указаны привязки различных типов компонентов к системам кодирования.

7.7.5.7 Примечания к реализации

Существуют отношения между типами компонентов и квалификаторами, которые могут с ними использоваться. Квалификаторы, которые могут использоваться с различными типами компонентов именования, указаны в таблице 20.

Таблица 20 — Отношения между типами компонентов и квалификаторами

	FAM (фамилия)	GIV (имя)	TITLE (обращение)	DEL (разделитель)	Пустое значение
LS (юридический статус)			√		√
AC (ученая степень или звание)			√		√
NB (дворянство)			√		√
PR (профессиональная принадлежность)			√		√
HON (почтительное)			√		√
BR (урожденное)	√	√			√
AD (приобретенное)	√	√			√
SP (супружеское)	√	√			√
MID (отчество, среднее имя)	√	√			√
CL (предпочтительное)	√	√			√
IN (инициал)	√	√			√
PFX (префикс)	√	√	√		√
SFX (суффикс)	√	√	√		√

√ — Это сочетание допустимо, но на практике вряд ли встретится.
√√ — Это сочетание допустимо и, скорее всего, будет встречаться на практике.

7.7.6 Тип данных EN (именование сущности)

7.7.6.1 Описание

Специализация типа данных ANY

Именование лица, организации, места или предмета.

Пример — «Джим Боб Уолтон, мл.», «Health Level Seven, Inc.», «Озеро Tahoe» и т. д. Именование объекта может быть простой строкой символов или состоять из нескольких компонентов, например, «Джим», «Боб», «Уолтон» и «мл.»; «Health Level Seven» и «Inc.»; «Озеро» и «Tahoe».

По существу, именования являются последовательностями своих компонентов, но при этом имеют дополнительные свойства use и useablePeriod, указывающие, когда именование использовалось и как выбрать одно из нескольких именований, действительных в одно и то же время.

7.7.6.2 Синтаксис ИСО/МЭК 11404

```
type EN = class (
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    part : Sequence(ENXP),
    use : Set(EntityNameUse)
)
```

7.7.6.3 Атрибуты

7.7.6.3.1 part : Sequence(ENXP): последовательность компонентов, например, имени или фамилии, префикса, суффикса.

7.7.6.3.2 use : Set(EntityNameUse): один или несколько кодов, информирующих систему или пользователя о том, какое из именований следует выбрать для данной цели.

Именование, для которого свойство use не указано, может быть по умолчанию использовано для любых целей, но именование с конкретным значением этого свойства должно предпочитаться для соответствующей цели. Именования не должны собираться без указания хотя бы одного кода использования use, но именования могут существовать без этого кода, особенно в унаследованных данных.

Если этому атрибуту присвоено значение, то оно должно быть взято из системы кодирования HL7 EntityNameUse. Текущие допустимые значения указаны в таблице 21.

Таблица 21 — Перечисление EntityNameUse, ОИД 2.16.840.1.113883.5.1120

Уровень	Код	Описание	Определение
1	Используемое представление: идентифицирует различные представления именования. Представление может влиять на использование именования (например, использование идеографического представления для формальных коммуникаций)		
2	ABC	Alphabetic (алфавитное)	Алфавитная транскрипция именования (romaji в Японии)
2	IDE	Ideographic (идеографическое)	Идеографическое представление именования (kanji в Японии, китайские иероглифы)
2	SYL	Syllabic (силлабическое)	Силлабическая транскрипция именования (например, kana в Японии, hangul в Корее)
1	C	Customary (привычное)	Привычное именование, которое обычно используют
1	OR	Official registry name	Формальное именование, зарегистрированное официальным (государственным) органом. Однако оно может использоваться не очень часто. Может соответствовать понятию юридически признаваемого наименования

Окончание таблицы 21

Уровень	Код	Имя	Определение
1	T	Temporary	Временное именование. Период его действия может предоставить более детальную информацию. Может также использоваться для временных именований, данных при рождении или при оказании экстренной помощи
1	Взятое именование: лицо взяло именование или ему дали его		
2	I	Indigenous/Tribal (племенное)	Например, «Вождь Красное Облако»
2	P	Other/ pseudonym/ alias (другое/ псевдоним/ вымышленное)	Неофициальное именование, под которым лицо иногда известно (оно может быть также использовано для регистрации неформальных именований, например, прозвищ)
2	ANON	Anonymous (анонимное)	Анонимное именование (используемое для защиты неприкосновенности личной жизни)
2	A	Business Name (деловое)	Именование, используемое в профессиональном или деловом контексте. ***Пример — Продолжение использования девичьей фамилии в профессиональном контексте или сценическое именование (некоторые из этих именований являются также псевдонимами)***
2	R	Religious (религиозное)	Именование, взятое как часть религиозного сана, например, «Сестра Мэри Браун», «Брат Джон»
1	OLD	No longer in use (более не используется)	Это именование более не используется. Примечание — Именования могут быть действительны в течение определенного срока. Этот код используется в ситуации, когда именование более не действительно, но конкретный срок, когда оно действовало, не известен
2	DN	Do not use (не использовать)	Это именование более не должно быть использовано при взаимодействии с данным лицом (например, не только не должно быть использовано, но даже не должно быть упомянуто при взаимодействии с этим лицом). Примечание — Не требуется, чтобы приложения сравнивали именования, помеченные «не использовать», с другими именованиями в целях исключения компонентов именования, общих для других именований и того, что помечено «не использовать»
1	M	Maiden name (девичье)	Именование, использовавшееся до вступления в брак. Обычаи супружеских именований широко варьируются по миру. Данное именование предназначено для приложений, которые собирают и хранят «девичьи» именования. Хотя понятие девичьего именования нередко имеет специфику пола, использование этого именования не специфично для пола. Оно не подразумевает ни конкретную историю изменений именования лица, ни возможность алгоритмического определения девичьего именования
1	SRCH	Search type uses (поисковое)	Именование, используемое для поиска или определения совпадения
2	PHON	Phonetic (фонетическое)	Именование, воспринимаемое оператором ввода, то есть близкая аппроксимация к произношению, не основанная на фонетическом алгоритме

Синтаксис ИСО/МЭК 11404 атрибута use:

```
type EntityNameUse = enumeration (C, OR, T, I, P, A, R, OLD, DN, M,
    SRCH, PHON, ABC, SYL, IDE)
```

Значения обоих атрибутов use и qualifier используются как множества, то есть можно указывать более одного значения каждого атрибута. Это может привести к синтаксически правильным, но семантически абсурдным сочетаниям кодов. Применяются следующие правила:

- одно именование сущности не может иметь более одного кода используемого представления;
- коды T, ABC, SYL и IDE рекомендуется использовать вместе с другим кодом использования именования;
- код квалификатора «LS» компонента именования организации не может сочетаться ни с каким другим квалификатором,. кроме PFX или SFX;
- квалификаторы BR и AD (или SP) взаимно несовместимы.

7.7.6.4 Равенство

Два значения именования равны, если их канонические формы содержат те же самые компоненты в том же самом порядке. Атрибуты use и validTime исключаются из проверки на равенство.

7.7.6.5 Инварианты

- Тип данных EN содержит атрибут nullFlavored или его часть.

Определение инвариантов на языке OCL:

```
inv "пусто или компоненты": isNull xor part-> notEmpty
```

7.7.6.6 Операции

7.7.6.6.1 canonical() : EN: каноническое именование со стандартным упорядочением компонентов.

Каноническая форма предложена в основном для целей определения равенства и может отличаться от общепринятого порядка представления компонентов именования в различных культурах по всему миру.

Каноническая форма содержит все типы компонентов, за исключением разделителей, в следующем порядке:

a) префиксы с квалификатором обращения;

b) имена с префиксами и/или суффиксами, ассоциированными с именами;

c) фамилии с префиксами и/или суффиксами, ассоциированными с фамилиями;

d) суффиксы с квалификатором обращения.

Каждый список типов компонентов должен быть упорядочен в соответствии с исходным именем.

7.7.6.7 Комментарии к технической спецификации ИСО/ТС 22220

Различные группы имен, определенные в технической спецификации ИСО/ТС 22220, непосредственно отображаются на типы компонентов ENXP. Компоненты использования именования и условного использования отображаются на атрибут use. Контекст использования типа данных EN может потребовать нескольких его экземпляров (на манер коллекции) для обеспечения всей функциональности, описанной в технической спецификации ИСО/ТС 22220.

7.7.6.8 Примеры

7.7.6.8.1 Простой пример

```
<example xsi:type="EN" >
    <part type="GIV" value="Adam"/>
    <part type="GIV" value="A."/>
    <part type="FAM" value="Everyman"/>
</example>
```

Очень простая кодировка именования Adam A. Everyman.

7.7.6.8.2 Сложный пример германского именования

```
<example xsi:type="EN.PN">
    <part type="GIV" qualifier="AC" value="Dr. phil."/>
    <part type="GIV" value="Regina"/>
```

```
    <part type="GIV" value="Johanna"/>
    <part type="GIV" value="Maria"/>
    <part type="TITLE" qualifier="PFX NB" value="Grafin"/>
    <part type="FAM" qualifier="BR" value="Hochheim"/>
    <part type="DEL" value=" "/>
    <part type="FAM" qualifier="SP" value="Weilenfels"/>
    <part type="TITLE" qualifier="SFX PR" value="NCFSA" />
</example>
```

Доктор философии Regina Johanna Maria Gräfin Hochheim-Weilenfels, NCFSA. Этот пример показывает широкое использование нескольких имен, префиксов и суффиксов ученой степени, дворянского титула и профессиональной принадлежности.

7.7.6.8.3 Наименование организации

```
<example xsi:type="EN.TN">
    <part value="Health Level Seven, Inc"/>
</example>
```

Наименование организации «Health Level Seven, Inc.» в простой строковой форме (тривиальное именование — EN.TN).

```
<example xsi:type="EN.ON">
    <part value="Health Level Seven, "/>
    <part type="TITLE" qualifier="SFX LS" value="Inc."/>
</example>
```

В качестве полностью разбираемого именования.

7.7.6.8.4 Японский пример

```
<example xsi:type="EN" use="IDE">
    <part type="FAM" value="木村"/>
    <part type="GIV" value="通男"/>
</example>
<example xsi:type="EN" use="SYL">
    <part type="FAM" value="きむら"/>
    <part type="GIV" value="みちお"/>
</example>
<example xsi:type="EN" use="ABC">
    <part type="FAM" value="KIMURA"/>
    <part type="GIV" value="MICHIO"/>
</example>
```

Японское именование в трех формах: идеографической (Kanji), силлабической (Hiragana) и алфавитной (Romaji).

7.7.6.8.5 Русский пример

```
<example xsi:type="EN">
    <part type="FAM" value="ЕМЕЛИН"/>
    <part type="GIV" value="ИВАН"/>
    <part type="GIV" value="ВЛАДИМИРОВИЧ"/>
</example>
<example xsi:type="EN">
    <part type="FAM" value="EMELIN"/>
    <part type="GIV" value="IVAN"/>
</example>
```

Русское именование в кириллице и в латинской транслитерации. В России эти именования известны соответственно как местное и заграничное. Системы должны определять конкретную используемую форму по набору символов компонентов именования.

7.7.6.8.6 Скандинавские примеры

```
<example xsi:type="EN" use="OR">
    <part type="GIV" value="Jan"/>
    <part type="GIV" value="Erik"/>
    <part type="FAM" qualifier="MID" value="Östlund"/>
    <part type="FAM" value="Erikson"/>
</example>
<example xsi:type="EN">
    <part type="GIV" value="Jan"/>
    <part type="FAM" value="Erikson"/>
</example>
```

Erikson является фамилией, Jan Erik — именами, а Östlund — фамилией матери, используемой как «Mellannamn».

```
<example xsi:type="EN" use="I">
<!-- Код использования мог быть OR+OLD в зависимости от того, как хранится
записье -->
    <part type="GIV" value="Margrete Jente"/>
    <part type="FAM" value="Hansen"/>
</example>
```

Jan Erikson и его жена Margrete Hansen имеют дочь Karin. В приведенном выше примере она сначала получила имя новорожденной «Margrete Jente» (дочь Margrete) и фамилию матери, а не имя Karin. Фамилия отца не использовалась. Это временное именование, присвоенное ребенку непосредственно при рождении.

```
<example xsi:type="EN" use="OR C">
    <part type="GIV" value="Karin"/>
    <part type="FAM" qualifier="MID" value="Hansen"/>
    <part type="FAM" value="Erikson"/>
</example>
```

Затем фамилия ребенка была заменена на фамилию отца, а фамилия матери стала использоваться как Mellomnamn.

```
<example xsi:type="EN" use="OR">
    <part type="GIV" value="Karin"/>
    <part type="FAM" qualifier="MID" value="Erikson"/>
    <part type="FAM" qualifier="SP" value="Berg"/>
</example>
<example xsi:type="EN" use="C">
    <part type="GIV" value="Karin"/>
    <part type="FAM" value="Berg"/>
</example>
```

Karin вышла замуж за Per Berg, решила принять Berg как фамилию, а Erikson использовать как Mellomnavn.

Примечание — Karin могла бы выбрать другую «среднюю фамилию» (Mellomnavn), например, фамилию своей матери, своего отца или другую фамилию в соответствии с юридическими правилами именований в ее стране.

7.7.6.8.7 Примеры прозвищ или неформальных именований

```
<example xsi:type="EN">
    <part type="GIV" value="Peter"/>
    <part type="GIV" qualifier="CL" value="James"/>
    <part type="FAM" value="Chalmers"/>
</example>
```

Полное именование Peter James Chalmers. Он предпочитает, чтобы его называли James (не «Jim» — нет, так его не называйте).

```
<example xsi:type="EN" use="OR">
    <part type="GIV" value="David"/>
    <part type="GIV" value="Woodford"/>
    <part type="FAM" value="Smith"/>
</example>
<example xsi:type="EN" use="C">
    <part type="GIV" value="Woody"/>
    <part type="FAM" value="Smith"/>
</example>
```

Его настоящее именование David Woodford Smith, но он предпочитает, чтобы его называли "Woody".

```
<example xsi:type="EN" use="OR">
    <part type="GIV" value="Uy"/>
    <part type="GIV" value="Dung"/>
    <part type="FAM" value="Nguyen"/>
</example>
<example xsi:type="EN" use="C">
    <part type="GIV" value="Dennis"/>
    <part type="FAM" value="Nguyen"/>
</example>
```

Урожденное именование лица "Uy Dung Nguyen", но после его иммиграции в западную страну он решил принять «вестернизованное» имя Dennis. Это обычная практика среди иммигрантов.

```
<example xsi:type="EN" use="OR C">
    <part type="GIV" value="Grahame"/>
    <part type="GIV" value="David"/>
    <part type="FAM" value="Grieve"/>
</example>
<example xsi:type="EN" use="P">
    <part type="GIV" value="Junior"/>
</example>
```

Урожденное именование лица "Grahame Grieve'. Обычно он его использует, но иногда его называют также "Junior".

7.7.6.8.8 Пример обращения

```
<example xsi:type="EN" use="OR C">
    <part type="TITLE" value="Dr"/>
    <part type="GIV" value="John"/>
    <part type="GIV" value="Paul"/>
    <part type="FAM" value="Jones"/>
    <part type="TITLE" qualifier="SFX" value="III"/>
    <part type="DEL" value=", "/>
```

```
    <part type="TITLE" qualifier="AC" value="PhD"/>
</example>
```

Полное именование Dr John Paul Jones III, PhD. Оно имеет код использования «OR», то есть официальное именование, но содержит обращения. Для целей настоящего стандарта обращения и разделители не считаются компонентами официального именования; однако они могут присутствовать, и нет никакой информации о том, включены ли они в официальный регистр.

Примечание — «Dr» является сокращением, а не инициалом для «Доктор», и не является инициалами. Инициалы могут содержать более одной буквы, если это требуется лингвистическими нормами соответствующего языка, но они не считаются сокращениями. Обращения часто сокращаются.

7.7.6.8.9 Сложные примеры

```
<example xsi:type="EN" use="OR C">
    <part type="GIV" value="Mary Jane"/>
    <part type="FAM" value="Contrata"/>
</example>
```

Mary Jane является двумя отдельными упорядоченными компонентами имени, разделенными пробелом, то есть «Jane» не представляет собой «среднее» имя.

Примечание — При автоматической генерации инициалов результат обычно должен иметь вид MC, а не MJC.

```
<example xsi:type="EN" use="OR C">
    <part type="GIV" value="Karen"/>
    <part type="FAM" value="Van"/>
    <part type="FAM" value="Hentenryck"/>
</example>
```

Karen Van Hentenryck имеет голландские корни, и «Van»' является приставкой (voorvoegsel).

```
<example xsi:type="EN" use="OR C">
    <part type="GIV" value="Selby"/>
    <part type="FAM" qualifier="SP" value="Butt"/>
    <part type="FAM" value="Beeler"/>
</example>
<example xsi:type="EN" use="OR OLD">
    <part type="GIV" value="Mary"/>
    <part type="FAM" qualifier="SP" value="Selby"/>
    <part type="FAM" value="Butt"/>
</example>
```

Урожденная Mary «Selby» Butt сменила при замужестве свое именование на Selby Butt Beeler, которое теперь записано в ее паспорте.

```
<example xsi:type="EN" use="OR A OLD">
    <part type="GIV" value="Jacqueline "/>
    <part type="GIV" value="Janette"/>
    <part type="GIV" value="Patricia"/>
    <part type="FAM" value="Campbell"/>
</example>
<example xsi:type="EN" use="P OLD">
    <part type="GIV" value="Ruth"/>
    <part type="FAM" value="Brinkman"/>
</example>
<example xsi:type="EN" use="P OLD">
```

```
    <part type="GIV" value="Ruth"/>
    <part type="FAM" qualifier="SP" value="Grieve "/>
</example>
<example xsi:type="EN" use="OR">
    <part type="GIV" value="Jacqueline"/>
    <part type="GIV" value="Janette"/>
    <part type="GIV" value="Patricia"/>
    <part type="FAM" value="Grieve"/>
</example>
<example xsi:type="EN" use="C">
    <part type="GIV" value="Jacque"/>
    <part type="FAM" value="Grieve"/>
</example>
<example xsi:type="EN" use="C OLD">
    <part type="GIV" value="Jacque Ruth"/>
    <part type="FAM" value="Grieve"/>
</example>
<example xsi:type="EN" use="M">
    <part type="FAM" value="Brinkman "/>
</example>
```

Это довольно сложный пример, но анонимизированный по отношению к реальному лицу. При рождении она получила именование "Jacqueline Janette Patricia Campbell", но росла под приемным именем "Ruth Brinkman". При выходе замуж ее стали называть "Ruth Grieve", но ее юридически признаваемое именование было "Jacqueline Janette Patricia Grieve". Позже она сменила свое имя на "Jacque-Ruth", а затем просто "Jacque". В качестве своей девичьей фамилии она пишет "Brinkman".

```
<example xsi:type="EN" use="OR OLD">
    <part type="GIV" qualifier="BR" value="Del-Roy"/>
    <part type="FAM" qualifier="BR" value="Burgess"/>
</example>
<example xsi:type="EN" use="P">
    <part type="GIV" value="Yor Led"/>
    <part type="FAM" value="Ssegrub"/>
</example>
<example xsi:type="EN" use="OR ABC OLD">
    <part type="GIV" qualifier="AD PFX" value="Abdul"/>
    <part type="DEL" value="-"/>
    <part type="GIV" qualifier="AD SFX" value="Malik"/>
    <part type="FAM" qualifier="AD" value="Shakir"/>
</example>
<example xsi:type="EN" use="OR ABC C">
    <part type="GIV" qualifier="AD" value="AbdulMalik"/>
    <part type="FAM" qualifier="AD" value="Shakir"/>
    <part type="TITLE" value="Sr"/>
</example>
<example xsi:type="EN" use="P DN">
    <part type="GIV" qualifier="AD" value="Abdul"/>
</example>
<example xsi:type="EN" use="P">
    <part value="AMS"/>
</example>
```

Другой сложный пример получен от реального лица, сообщившего, что при рождении ему было дано именование Del-Roy Burgess, а его прозвищем было Yor-Led Ssegrub. Приняв ислам, он изменил свое именование на Abdul-Malik Shakir. Его написание соответствует арабскому имени в латинском

алфавите. Если Abdul-Malik слишком длинно, называйте его AMS, но не Abdul. Недавно он начал писать свое имя в нотации «верблюжьего горба», опуская дефис (то есть AbdulMalik вместо Abdul-Malik). Кроме того, он начал использовать суффикс «Sr.», чтобы отличить себя от его сына, названного тем же именем. Этот суффикс используется нечасто, за исключением нескольких официальных регистров, например, в паспорте, водительских правах и других документах, где важна однозначная идентичность.

7.7.7 Тонкость EN.TN (тривиальное именование)

7.7.7.1 Описание

Ограничение типа данных EN

Ограничение типа данных EN, представляющее собой простую строку, используемую в качестве простого именования предметов и мест. Такие именования обычно используются для предметов и мест, например, «Озеро Эри» или «Национальный аэропорт Вашингтона — Рейгана».

7.7.7.2 Инварианты

- если тонкость EN.TN не пуста, то она может иметь только один компонент, у которого нет атрибутов type или qualifier.

Определение инвариантов на языке OCL:

```
inv "только один компонент без типа ": isNotNull implies
    (part->size = 1 and part->first.type.oclIsUndefined and
    part->first.qualifier->isEmpty)
```

7.7.8 Тонкость EN.PN (именование лица)

7.7.8.1 Описание

Ограничение типа данных EN

Ограничение типа данных EN, используемое, когда именованной сущностью является физическое лицо. Представляет последовательность компонентов именования, например, имя или фамилию, префикс, суффикс.

В этом ограничении допускает использовать компоненты именования только с теми квалификаторами, которые применимы к именованиям физических лиц.

Примечание — Поскольку структура именования сущности рассчитана в основном на требования, предъявляемые к именованиям физических лиц, то ограничение является достаточно слабым.

7.7.8.2 Инварианты

- компоненты именования физического лица не могут иметь квалификатор LS (юридический статус).

Определение инвариантов на языке OCL:

```
inv "компоненты не могут иметь квалификатор LS ": part->forAll(p | not
    p.qualifier->includes(EntityNamePartQualifier.LS))
```

7.7.9 Тонкость EN.ON (наименование организации)

7.7.9.1 Описание

Ограничение типа данных EN

7.7.9.2 Инварианты:

- компоненты наименования организации не могут иметь тип FAM или GIV;
- следующие квалификаторы не должны использоваться в наименовании организации: I, P, ANON, A, R, DN и M.

Определение инвариантов на языке OCL:

```
inv "компоненты не должны иметь типы, относящиеся к физическим лицам": part-
>forAll(p |
    not (p.type = EntityNamePartType.FAM or
    p.type = EntityNamePartType.GIV))
inv "коды использования не должны иметь значения, относящиеся к физическим лицам ":
    use->forAll(u | not
    (u = EntityNameUse.I
    or u = EntityNameUse.P
```

```
    or u = EntityNameUse.ANON
    or u = EntityNameUse.A
    or u = EntityNameUse.R
    or u = EntityNameUse.M)).M))
```

7.8 Количественные типы данных

7.8.1 Введение

Эти типы данных предназначены для представления физических величин (см. рисунок 7).

7.8.2 Тип данных QTY (количество)

7.8.2.1 Описание

Специализация типа данных ANY

Тип данных количества представляет собой абстрактную генерализацию всех типов данных, у которых домен значений упорядочен (отношением «меньше или равно») и при этом определена операция вычитания для всех полностью упорядоченных подмножеств этого домена.

Абстракция типа данных количества необходима для определения некоторых других типов, например, интервал и распределение вероятностей.

7.8.2.2 Синтаксис ИСО/МЭК 11404

```
type QTY = class {
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    expression : ED,
    originalText : ED.TEXT,
    uncertainty : QTY,
    uncertaintyType : UncertaintyType
    uncertainRange : IVL(QTY)
}
```

В типе данных QTY определены три свойства, которые могут иметь все величины: выражение (expression), которое может использоваться для вычисления фактической величины, исходный текст (originalText), в которому указана исходная форма представления величины, и неопределенность (uncertainty) величины. Существуют две различные формы представления неопределенности: статистическая, обычно пригодная для измеряемых величин, и форма диапазона, которая обычно ассоциируется с указаниями (например, принимать от четырех до шести таблеток).

Наличие этих атрибутов может значительно усложнить правильное толкование значения. Поэтому их использование должно строго контролироваться во всех контекстах использования. Если в элементах обработки информации используются величины, то в объявлениях соответствия должно быть точно указано, как и когда используются эти атрибуты.

7.8.2.3 Атрибуты

7.8.2.3.1 expression : ED: выражение, которое может использоваться для получения фактического значения количественной информации, взятой из контекста использования.

Например, выражение может использоваться в инструкции по дозировке, зависящей от массы тела пациента.

Если значение типа QTY не задает правильное количество, то вне зависимости от того, задано выражение или нет, оно должно иметь причину пустоты nullFlavor. Если правильное количество не задано, а выражение присутствует, то обычно причина пустоты имеет значение DER. Если указаны и правильное количество, и выражение, то причина пустоты не требуется; в этих случаях процесс обработки должен сам определить, когда должно вычисляться выражение.

Язык выражения наследуется от значения атрибута mediatype (таблица 22). Если для выражения представлено несколько переводов, то вычислитель может сам выбрать предпочтительный язык; все переводы должны приводить к одному и тому же результату.

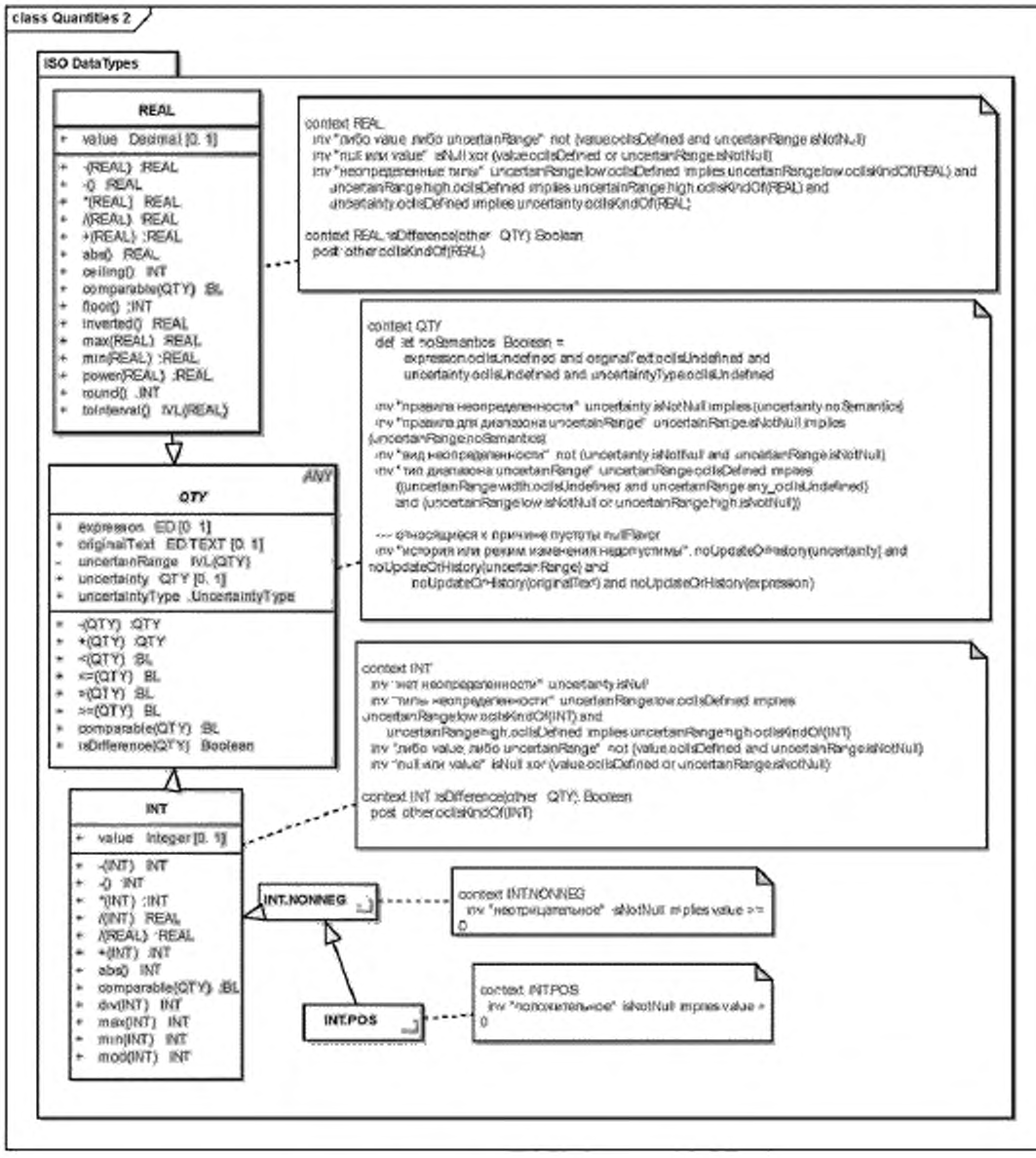


Рисунок 7, лист 1 — Количественные типы данных

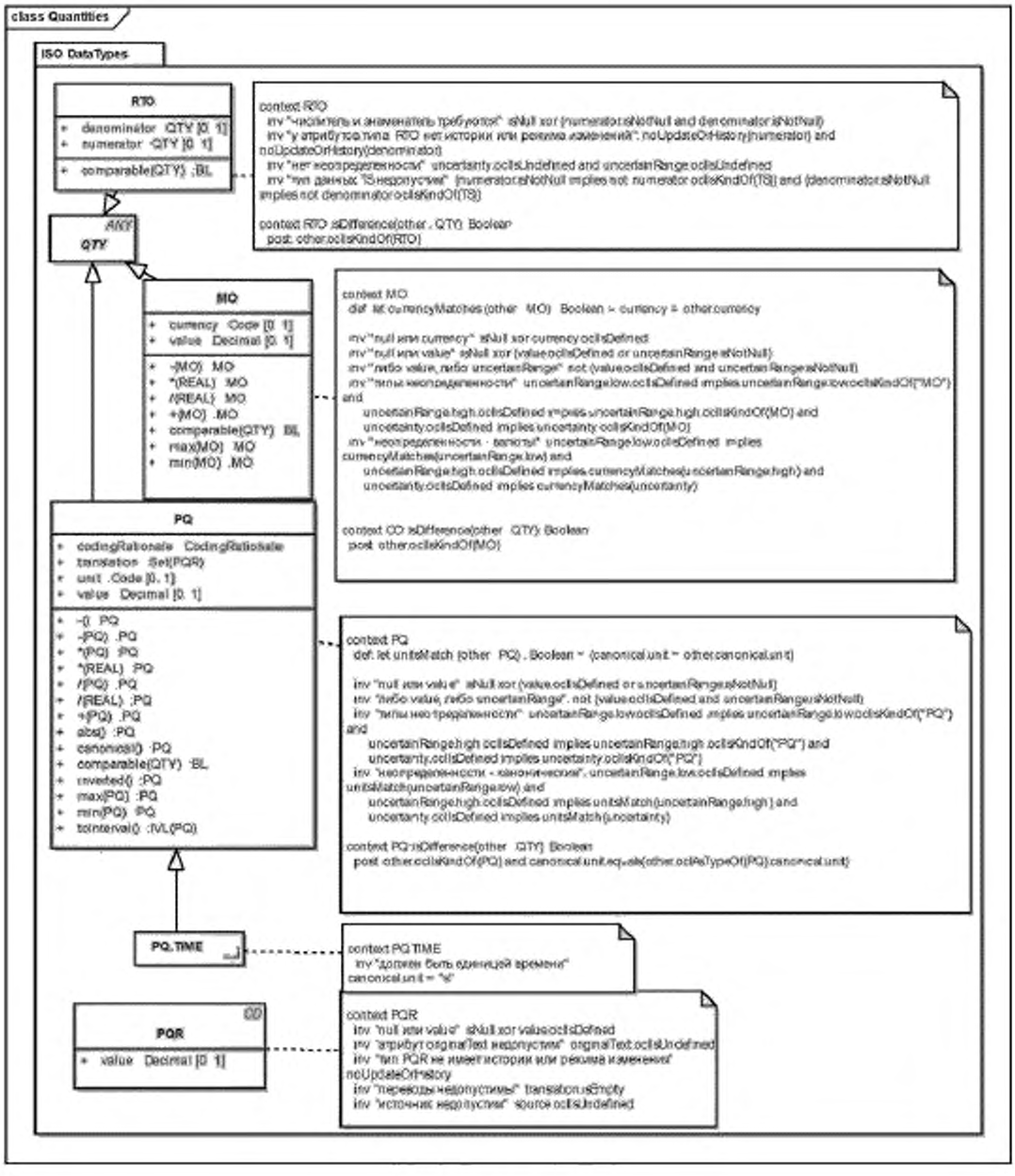


Рисунок 7, лист 2

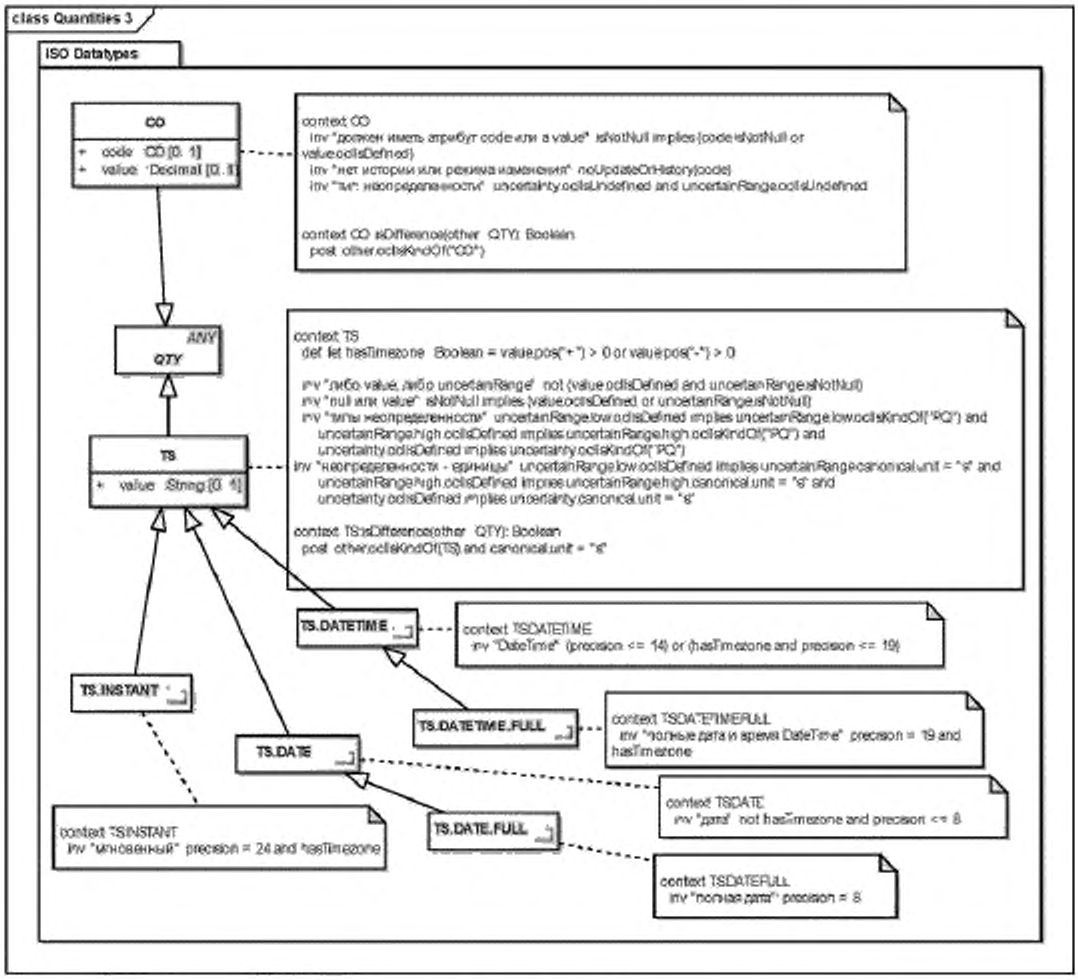


Рисунок 7, лист 3

Язык определяет формы, которые может принимать свойство expression. Он определяет также, как информация из контекста выражения может быть доступна языковыми средствами и как определяется на нем новая форма значения. Языки могут использоваться только в том случае, если эта информация правильным образом определена для контекста использования типа данных QTY.

От элементов обработки информации не требуется, чтобы для объявления непосредственного или косвенного соответствия настоящему стандарту ими были реализованы все языки, но в их объявлениях соответствия должно быть указано, какие языки ими поддерживаются.

Таблица 22 — Соответствие языка выражения и типа среды mediatype

Язык	Тип среды mediatype
OCL	text/plain+ocl
Factor	application/hl7-factor+xml
MathML	application/mathml+xml
Примечание — Язык Factor документирован в спецификации HL7 Abstract Data Types.	

7.8.2.3.2 originalText : ED.TEXT: исходный текст, на основе которого кодируется значение типа QTY (если он служит источником этого значения).

Исходный текст может использоваться в структурированном интерфейсе пользователя, чтобы описать, что видел пользователь на экранной форме в качестве представления величины, либо в ситуации, когда пользователь диктует или непосредственно вводит текст. Этот тот текст, который введен или произнесен пользователем.

Допускает использовать потомки типа данных QTY для хранения только исходного текста. В этом случае исходный текст существует без правильного значения величины. В ситуации, когда значение величины определяется через какое-то время после ввода текста, атрибут originalText содержит текст или фразу, используемые в качестве основы определения этого значения. Содержание атрибута originalText не заменяет правильное значение. Если фактическое значение типа QTY не является правильным, то для этого значения должна быть указана причина пустоты nullFlavor вне зависимости от того, имеет атрибут originalText значение или нет.

Исходный текст должен представлять собой выдержку из релевантной информации, взятой из оригинальных источников, а не указатель на нее либо ее полное воспроизведение. Поэтому исходный текст должен представляться в неформатированном виде. В специфичных обстоятельствах, когда контекст использования точно описан, атрибут исходного текста originalText может быть ссылкой на некоторый другой текстовый артефакт, для которого однозначно определены способы разрешения ссылки.

Примечание — Детали связывания ссылки originalText.reference с различными артефактами медицинской информации (например, с документом и кодированным результатом) не входят в область применения настоящего стандарта и могут быть предписаны в спецификациях, использующих настоящий стандарт.

7.8.2.3.3 uncertainty : QTY: неопределенность величины, указанная с помощью функции распределения и ее параметров. Это основная мера отклонения/неопределенности величины (квадратный корень из суммы квадратов разности между всеми точками данных и средней величиной). Фактический тип неопределенности зависит от конкретного потомка типа данных QTY и фиксирован для каждого потомка.

Существуют два разных вида представления неопределенности. Данный вид в сочетании с атрибутом uncertaintyType представляет статистическую неопределенность. Диапазон неопределенности, указываемый в атрибуте uncertainRange, описывает иной вид неопределенности, не подразумевающий статистическое распределение.

Эта форма неопределенности может быть применима только к доменам с непрерывными значениями (REAL, PQ, MO и TS). В типе данных RTO неопределенность может быть указана отдельно для числителя и отдельно для знаменателя.

У неопределенности должен отсутствовать атрибут expression. Неопределенности не могут содержать другие неопределенности. Атрибут originalText не может быть указан для неопределенности — любая неопределенность, ассоциируемая со значением типа QTY, должна вытекать из содержания атрибута originalText этого значения.

Неопределенность не может иметь собственный атрибут originalText, поскольку ожидается, что неопределенность величины должна быть представлена в содержании атрибута originalText самой величины.

7.8.2.3.4 uncertaintyType : UncertaintyType: код, идентифицирующий тип распределения вероятности неопределенности.

Существуют два разных вида представления неопределенности. Сочетание атрибута uncertainty с атрибутом uncertaintyType представляет статистическую неопределенность. Диапазон неопределенности, указываемый в атрибуте uncertainRange, описывает иной вид неопределенности, не подразумевающий статистическое распределение.

Пустое (неизвестное) значение кода типа распределения указывает, что тип распределения вероятности неизвестен. Если при этом атрибут uncertainty имеет значение, то оно является эмпирической оценкой меры неопределенности.

Если этому атрибуту присвоено значение, то оно должно быть взято из системы кодирования HL7 DistributionType. Текущие допустимые значения указаны в таблице 23.

Таблица 23 — Перечисление UncertaintyType, OID: 2.16.840.1.113883.5.1020

Уровень	Код	Описание	Определение
1	U	Uniform (равномерное)	Равномерное распределение задает постоянную вероятность всему интервалу возможных результатов, а все результаты вне этого интервала имеют нулевую вероятность. Длина интервала равна $2\ \sigma\ \sqrt{3}$. Таким образом, равномерное распределение имеет плотность вероятности $f(x) = (2\ \sigma\ \sqrt{3})^{-1}$ для значений x в интервале $\mu - \sigma\ \sqrt{3} \geq x \leq \mu + \sigma\ \sqrt{3}$ и $f(x) = 0$ для остальных значений x
1	N	Normal (gaussian) (нормальное, Гауссово)	Это хорошо известное нормальное распределение в форме колокола. В соответствии с центральной предельной теоремой нормальное распределение представляет собой распределение неограниченной случайной величины, являющееся результатом суммы большого количества стохастических процессов. Нормальное распределение может быть достаточно точным даже для величин, значения которых ограничены с одной стороны (например, больше 0), при условии, что среднее значение находится «достаточно далеко» от этой границы (в терминах стандартного отклонения)
1	LN	Log-normal (логарифмическое нормальное распределение)	Логарифмическое нормальное распределение используется для преобразования асимметричной случайной величины в нормально распределенную случайную величину $U = \log X$. Логарифмическое нормальное распределение может быть задано свойствами среднего значения μ и стандартного отклонения σ. Следует учесть, что среднее значение μ и стандартное отклонение σ являются параметрами исходного распределения, а не преобразованными параметрами логарифмического нормального распределения, которые традиционно обозначаются теми же буквами. Эти параметры μ_{log} и σ_{log} связаны со средним значением μ и стандартным отклонением значения данных соотношениями $\sigma_{log}^2 = \log(\sigma^2/\mu^2 + 1)$ и $\mu_{log} = \log \mu - \sigma_{log}^2/2$
1	G	(gamma) (гамма)	Гамма-распределение используется для асимметричных данных, ограниченных справа, для которых максимум кривой распределения находится недалеко от начала координат. Гамма-распределение имеет два параметра α и β. Их соотношения со средним значением μ и отклонением σ^2 имеют вид $\mu = \alpha\ \beta$ и $\sigma^2 = \alpha\ \beta^2$
1	E	Exponential (экспоненциальное)	Используется для данных, описывающих наработку на отказ. Экспоненциальное распределение является частным случаем γ-распределения, у которого $\alpha = 1$, следовательно, соотношения его параметра со средним значением μ и отклонением σ^2 имеют вид $\mu = \beta$ и $\sigma^2 = \beta^2$
1	χ^2	Хи-квадрат распределение	Используется для описания суммы квадратов случайных величин, возникающей, когда оценивается (а не предполагается) отклонение экспериментальных результатов. Единственным параметром распределения χ^2 является υ, так называемое число степеней свободы (равное числу независимых компонентов суммы). Распределение χ^2 — частным случаем γ-распределения, у которого параметр $\alpha = \upsilon/2$, а параметр $\beta = 2$. Следовательно, $\mu = \upsilon$ и $\sigma^2 = 2\ \upsilon$
1	T	*t* (Student) (распределение Стьюдента)	Используется для описания отношения случайной величины с нормальным распределением к квадратному корню из случайной величины с распределением χ^2. Распределение Стьюдента имеет один параметр υ, число степеней свободы. Соотношения его параметра со средним значением μ и отклонением σ^2 имеют вид $\mu = 0$ и $\sigma^2 = \upsilon / (\upsilon - 2)$

Окончание таблицы 23

Уровень	Код	Описание	Определение
1	F	*F*-распределение	Используется для описания отношения двух случайных величин с распределением χ^2. *F*-распределение имеет два параметра υ_1 и υ_2, равные числу степеней свободы соответственно числителя и знаменателя. Соотношения этих параметров со средним значением μ и отклонением σ^2 имеют вид $\mu = \upsilon_2 / (\upsilon_2 - 2)$ и $\sigma^2 = (2\ \upsilon_2^2\ (\upsilon_2 + \upsilon_1 - 2)) / (\upsilon_1\ (\upsilon_2 - 2)^2\ (\upsilon_2 - 4))$
1	B	β (beta) (Бета распределение)	Бета-распределение используется для данных, которые ограничены с двух сторон и могут быть, а могут и не быть асимметричными (что, к примеру, имеет место при оценке вероятности). Оно имеет два параметра α и β. Соотношения этих параметров со средним значением μ и отклонением σ^2 имеют вид $\mu = \alpha / (\alpha + \beta)$ и $(\sigma^2 = \alpha\ \beta/((\alpha + \beta)^2\ (\alpha + \beta + 1))$

Многие типы распределений определены в терминах специальных параметров (например, параметры α и β для γ-распределения, число степеней свободы для *t*-распределения). Но для всех типов распределений определены среднее значение и стандартное отклонение.

Если у атрибута distributionType нет значения (null), то оценка среднего значения производится без какого-либо предположения о распределении вероятности значений. В этом случае смысл стандартного отклонения не имеет твердого определения. Однако можно следовать в русле интерпретации, принятой для нормального распределения, например, интервал «среднее значение ±1 стандартное отклонение» имеет доверительный уровень примерно две трети.

Каждая система, объявляющая о реализации неопределенности, должна поддерживать три типа распределения: неизвестное (null), равномерное и нормальное. Все другие типы распределений необязательны. Если система, интерпретирующая представление неопределенности, встречается с типом распределения, которое она не может распознать, то она должна отображать его на тип неизвестного распределения (null).

Синтаксис ИСО/МЭК 11404 атрибута uncertaintyType:

```
type UncertaintyType = enumeration (U, N, LN, G, E, X2, T, F, B)
```

7.8.2.3.5 uncertainRange . IVL(QTY): указывает, что значение принадлежит определенному интервалу возможных значений.

Атрибут uncertainRange используется, если фактическое значение неизвестно, но можно утверждать, что оно принадлежит определенному интервалу возможных значений. Атрибут uncertainRange отличается от атрибута uncertainty тем, что последний используется для указания конкретного значения и распределения его неопределенности либо для передачи информации о распределении множества значений, в то время как атрибут uncertainRange указывает, что существует единственное значение, которое хотя и неизвестно, но принадлежит определенному интервалу значений. При этом никаких предположений о распределении значений в этом интервале не делается.

Значение атрибута uncertainRange нередко ассоциируется с указанием выполнить определенную операцию в некоторый момент заданного интервала времени.

Если атрибут uncertainRange имеет значение, то должны быть заданы его свойства low (нижняя граница) или high (верхняя граница). Свойства any и width интервального типа данных IVL не могут использоваться. Если атрибут uncertainRange имеет значение, то атрибут value не может иметь значение.

7.8.2.4 Равенство

Поскольку тип данных QTY является абстрактным, то равенство его экземпляров не определено. Атрибуты типа данных QTY (expression, originalText, uncertainty и uncertaintyType) никогда не участвуют в определении равенства значений специализаций типа данных QTY.

7.8.2.5 Инварианты:

- у атрибута неопределенности (uncertainty) нет свойств expression, uncertainty или originalText;
- у атрибута интервала неопределенности (uncertainRange) нет свойств expression, uncertainty или originalText;
- атрибуты uncertainty и uncertainRange не могут одновременно иметь значение;
- атрибут uncertainRange не может иметь свойство width или any.

Определение инвариантов на языке OCL:

```
def: let noSemantics : Boolean = expression.oclIsUndefined and
    originalText.oclIsUndefined and uncertainty.oclIsUndefined and
    uncertaintyType.oclIsUndefined
inv "правила для атрибута uncertainty ": uncertainty.isNotNull implies
    (uncertainty.noSemantics)
inv " правила для атрибута uncertainRange": uncertainRange.isNotNull implies
    (uncertainRange.noSemantics)
inv "вид неопределенности": not (uncertainty.isNotNull and
    uncertainRange.isNotNull)
inv "тип интервала неопределенности": uncertainRange.oclIsDefined implies
    ((uncertainRange.width.oclIsUndefined and
    uncertainRange.any.oclIsUndefined)
    and (uncertainRange.low.isNotNull or
    uncertainRange.high.isNotNull))
inv "нет истории или режима изменения": noUpdateOrHistory(uncertainty) and
    noUpdateOrHistory(uncertainRange) and
    noUpdateOrHistory(originalText) and
    noUpdateOrHistory(expression))
```

7.8.2.6 Операции

7.8.2.6.1 lessThan[<](other : QTY) : BL: имеет значение true, если текущее значение (this) меньше другого значения (other). Для неопределенных значений результат операции может быть неизвестен (result = NullFlavor.UNK).

7.8.2.6.2 lessOrEqual[<=](other : QTY) : BL: имеет значение true, если текущее значение (this) меньше другого значения (other) или равно ему. Для неопределенных значений результат операции может быть неизвестен (result = NullFlavor.UNK).

7.8.2.6.3 greaterOrEqual[>=](other : QTY) : BL: имеет значение true, если текущее значение (this) больше другого значения (other) или равно ему. Для неопределенных значений результат операции может быть неизвестен (result = NullFlavor.UNK).

7.8.2.6.4 greaterThan[>](other : QTY) : BL: имеет значение true, если текущее значение (this) больше другого значения (other). Для неопределенных значений результат операции может быть неизвестен (result = NullFlavor.UNK).

7.8.2.6.5 plus[+](other . QTY) : QTY: результат сложения текущего значения (this) и другого значения (other). Другое значение должно иметь правильный тип данных (тот же, что у текущего значения, за исключением типа данных TS, для которого текущее значение должно иметь тип данных PQ с единицами времени, и типа данных PQ, для которого другое значение должно иметь совместимые единицы измерения), в противном случае результат имеет значение причины пустоты NI. Неопределенности должны переноситься на результат операции. Если значения имеют смешанные типы неопределенности, то результат может быть неизвестен (result = NullFlavor.UNK).

7.8.2.6.6 minus[-](other : QTY) : QTY: результат вычитания другого значения (other) из текущего значения (this). Другое значение должно иметь правильный тип данных (тот же, что у текущего значения, за исключением типа данных TS, для которого текущее значение должно иметь тип данных TS или PQ с единицами времени, и типа данных PQ, для которого другое значение должно иметь совместимые единицы измерения), в противном случае результат имеет значение причины пустоты NI. Неопределенности должны переноситься на результат операции. Если значения имеют смешанные типы неопределенности, то результат может быть неизвестен (result = NullFlavor.UNK).

7.8.2.6.7 comparable(other : QTY) : Boolean: указывает, можно ли сравнивать текущее значение (this) и другое значение (other) в операции равенства.

Примечание — Обычно эта операция возвращает значение true, если текущее значение (this) и другое значение (other) имеют одинаковый тип данных, если иное не описано для специализаций типа данных QTY.

7.8.2.6.8 isDifference(other : QTY) :BL: имеет значение true, если значение other является экземпляром, представляющим разность между двумя экземплярами этого типа.

Примечание — Обычно эта операция возвращает значение true, если другое значение (other) того же типа, за исключением типа данных TS, для которого разность имеет тип данных PQ с единицами, являющимися разновидностями единиц времени, и типа данных PQ, для которого единицы измерения должны быть совместимы.

7.8.3 Тип данных INT (целочисленный)

7.8.3.1 Описание

Специализация типа данных QTY

Целые числа (−1,0,1,2, 100, 3398129, etc.) являются точными величинами, являющимися результатами подсчетов и нумерации. Целые числа дискретны, множество целых чисел бесконечно, но счетно. На диапазон целых чисел никакие произвольные ограничения не накладываются.

7.8.3.2 Синтаксис ИСО/МЭК 11404

```
type INT = class {
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    expression : ED,
    originalText : ED.TEXT,
    uncertainty : QTY,
    uncertaintyType : UncertaintyType,
    uncertainRange : IVL(QTY)
    value : integer
}
```

7.8.3.3 Атрибуты

value : Integer: значение экземпляра типа INT. Настоящий стандарт не накладывает никаких ограничений на размер целого числа, но большинство реализаций ограничиваются 32-битовыми или 64-битовыми целыми числами.

Это пример шаблона обертки примитивного типа данных. Дополнительные сведения приведены в 6.3.

7.8.3.4 Равенство

Два непустых экземпляра типа INT равны, если они не имеют причин пустоты и имеют одно и то же значение либо их интервалы неопределенности uncertainRange не пусты, не имеют причины пустоты nullFlavor и равны.

7.8.3.5 Инварианты:

- если атрибуты value или uncertainRange не имеют причины пустоты nullFlavor, то они должны иметь значение;
- атрибуты value и uncertainRange не могут одновременно иметь значение;
- атрибут uncertainRange должен иметь тип IVL(INT);
- атрибут uncertainty не должен иметь значение.

Определение инвариантов на языке OCL:

```
inv "нет неопределенности": uncertainty.isNull
inv "типы неопределенности": uncertainRange.low.oclIsDefined
    implies uncertainRange.low.oclIsKindOf(INT) and
    uncertainRange.high.oclIsDefined implies
    uncertainRange.high.oclIsKindOf(INT)
inv "либо value, либо uncertainRange": not (value.oclIsDefined and
    uncertainRange.isNotNull)
inv "null либо value": isNull xor (value.oclIsDefined or
    uncertainRange.isNotNull)
inv "null либо value": isNull xor value.oclIsDefined
```

7.8.3.6 Операции

7.8.3.6.1 negated[−] . INT: изменение знака текущего значения (this).

7.8.3.6.2 plus[+] (other : INT) : INT: значение суммы текущего значения (this) и другого значения (other).

7.8.3.6.3 minus[−] (other : INT) : INT: значение разности текущего значения (this) и другого значения (other).

7.8.3.6.4 times[*] (other : INT) : INT: значение произведения текущего значения (this) на другое значение (other).

7.8.3.6.5 dividedBy[/] (other : INT) : REAL: значение деления текущего значения (this) на другое значение (other). Если другое значение равно 0, то результат имеет причину пустоты nullFalvor с кодом NI.

7.8.3.6.6 dividedBy[/] (other : REAL) : REAL: значение деления текущего значения (this) на другое значение (other). Если другое значение равно 0, то результат имеет причину пустоты NI.

7.8.3.6.7 abs() : INT: абсолютная величина текущего значения (this).

7.8.3.6.8 div(other : INT) : INT: целочисленное деление текущего значения (this) на другое значение (other).

7.8.3.6.9 mod(other : INT) : INT: остаток от целочисленного деления текущего значения (this) на другое значение (other).

7.8.3.6.10 max(other : INT) : INT: максимум из текущего значения (this) и другого значения (other).

7.8.3.6.11 min(other : INT) : INT: минимум из текущего значения (this) и другого значения (other).

7.8.3.6.12 comparable(other : QTY) : BL: целые числа всегда можно сравнить.

7.8.3.7 Примеры

7.8.3.7.1 Правильное значение

```
<example xsi:type="INT" value="23"/>
```

Целое число 23.

7.8.3.7.2 Неизвестное значение

```
<example xsi:type="INT" nullFlavor="NASK"/>
```

Пациентку не спросили об этом значении. Например, она никогда не была беременной, поэтому ее не спросили о том, сколько у нее детей.

7.8.4 Тонкость INT.NONNEG

7.8.4.1 Описание

Ограничение типа данных INT

Тонкость NT.NONNEG ограничивает тип данных INT неотрицательными числами.

7.8.4.2 Инварианты

- если значение не имеет причины пустоты nullFlavor, то оно должно быть неотрицательным и не иметь неопределенности.

Определение инвариантов на языке OCL:

```
inv "неотрицательное": isNotNull implies value >= 0
```

7.8.5 Тонкость INT.POS

7.8.5.1 Описание

Ограничение тонкости INT.NONNEG

Тонкость INT.POS ограничивает тонкость INT.NONNEG положительными числами.

7.8.5.2 Инварианты:

- если значение не имеет причины пустоты nullFlavor, то оно должно быть положительным и не иметь неопределенности.

Определение инвариантов на языке OCL:

```
inv "положительное": isNotNull implies value > 0
```

7.8.6 Тип данных CO (кодированное порядковое число)

7.8.6.1 Описание

Специализация типа данных QTY

Представляет данные, в которых кодированные значения ассоциируются с определенным упорядочением.

Тип данных CO может использоваться для сущностей, моделирующих ранжирование и шкалы, например, шкала Ликерта, шкала боли, шкала Апгар, в которых: а) подразумевается упорядочение; б) не предполагается, что расстояние между последовательными значениями постоянно; в) общее число значений конечно. Тип данных CO может также использоваться в контексте упорядоченной системы кодирования. В этом случае использование атрибута value может оказаться неподходящим или даже невозможным, но тип данных CO тем не менее может быть использован в модели, позволяющей с помощью таких систем кодирования описывать элементы, характеризующие порядок вхождения терминов в домен значений.

Относительный порядок значений в системе кодирования может не соответствовать лексикографическому упорядочению литерального представления значений типа CO. В этих случаях предполагается, что приложение определит упорядочение этих значений по некоторому определению системы кодирования.

В некоторых системах кодирования понятиям непосредственно присвоены числовые значения, с которыми можно выполнять математические операции.

Хотя в принципе это могло бы иметь смысл, приложения не должны предполагать, что переводы кодов, если таковые имеются, будут иметь тот же порядок, что и сами значения типа CO. Переводы не должны учитываться при определении упорядочения системы кодирования.

7.8.6.2 Синтаксис ИСО/МЭК 11404

```
type CO = class {
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    expression : ED,
    originalText : ED.TEXT,
    uncertainty : QTY,
    uncertaintyType : UncertaintyType,
    uncertainRange : IVL(QTY)
    value : Decimal,
    code : CD
}
```

7.8.6.3 Атрибуты

7.8.6.3.1 value : Decimal: числовое значение, ассоциированное с кодированным порядковым значением.

В некоторых контекстах использования это значение может быть ограничено целыми числами. Если атрибут code имеет непустое значение, то атрибут value должен быть непустым только в том случае, если система кодирования явно присвоила его значение данному понятию.

7.8.6.3.2 code : CD: код, представляющий определение порядкового элемента.

7.8.6.4 Равенство

Два непустых значения типа CO равны, если равны их атрибуты code.

Примечания

1 Значения типа CO, у которых атрибуты value присутствуют, а атрибуты code отсутствуют, никогда не равны, поскольку непонятно, являются ли они сопоставимыми порядковыми значениями.

2 Поскольку определение равенства значений типа CO основано на равенстве их атрибутов code, то значения типа CO могут быть равны значениям типа CD.

7.8.6.5 Инварианты:

- если значение не имеет причины пустоты nullFlavor, то оно должно иметь или атрибут code, или атрибут value;

- неопределенность не допускается.

Определения инвариантов на языке OCL:

```
inv "должно иметь код или значение": isNotNull implies
(code.isNotNull or value.oclIsDefined)
        inv "тип неопределенности ": uncertainty.oclIsUndefined and
              uncertainRange.oclIsUndefined
  inv "нет истории или режима изменений":noUpdateOrHistory(code)
```

7.8.6.6 Операции

7.8.6.6.1 max(other : CO) : CO: максимум из текущего значения (this) и другого значения (other).

Примечание — Если атрибут value не указан, то для определения порядка двух значений может понадобиться обращение к соответствующей терминологии. Если порядок не определен, то результат должен иметь причину пустоты nullFlavor с кодом NI.

7.8.6.6.2 min(other : CO) : CO: минимум из текущего значения (this) и другого значения (other).

Примечание — Если атрибут value не указан, то для определения порядка двух значений может понадобиться обращение к соответствующей терминологии. Если порядок не определен, то результат должен иметь причину пустоты nullFlavor с кодом NI.

7.8.6.6.3 comparable(other : QTY) :BL : имеет значение true, только если текущее значение (this) и другое значение (other) принадлежат к одной и той же упорядоченной системе кодирования codeSystem. В противном случае эта операция имеет значение false.

7.8.6.6.4 plus[+](other : QTY) : QTY: эта операция возвращает непустое значение, только если this. comparable(other) равно true и в системе кодирования codeSystem имеет смысл операция сложения этих кодов. В противном случае эта операция имеет значение false.

7.8.6.6.5 minus[-](other : QTY) : QTY: эта операция возвращает непустое значение, только если this. comparable(other) равно true и в системе кодирования codeSystem имеет смысл операция вычитания этих кодов. В противном случае эта операция имеет значение false.

7.8.6.7 Примеры

```
<example xsi:type="CO" value="1">
  <code code="1" codeSystem="2.16.840.1.113883.2.6.15.1.1">
    <displayName value="Poor"/>
  </code>
</example>
```

В данном случае значению «poor» присвоено числовое значение 1.

7.8.7 REAL (вещественное число)

7.8.7.1 Описание

Специализация типа данных QTY

Дробные числа. Обычно используются, когда величины измеряются, оцениваются или вычисляются из других вещественных чисел. Типичным представлением является десятичная запись, в которой число значащих десятичных цифр называется точностью.

7.8.7.2 Синтаксис ИСО/МЭК 11404

```
type REAL = class {
  validTimeLow : characterstring,
  validTimeHigh : characterstring,
  controlInformationRoot : characterstring,
  controlInformationExtension : characterstring,
  nullFlavor : NullFlavor,
  updateMode : UpdateMode,
  flavorId : Set(characterstring),
  expression : ED,
  originalText : ED.TEXT,
```

```
    uncertainty : QTY,
    uncertaintyType : UncertaintyType,
    uncertainRange : IVL(QTY)
    value : Decimal
)
```

7.8.7.3 Атрибуты

7.8.7.3.1 value : Decimal: значение типа REAL.

Это пример шаблона обертки примитивного типа данных. Дополнительные сведения приведены в 6.3.

7.8.7.4 Равенство

Два непустых экземпляра типа REAL равны, если они не имеют причин пустоты и имеют одно и то же значение, либо их интервалы неопределенности uncertainRange не пусты, не имеют причины пустоты nullFlavor и равны.

7.8.7.5 Инварианты:

- если атрибуты value или uncertainRange не имеют причины пустоты nullFlavor, то они должны иметь значение;
- атрибуты value и uncertainRange не могут одновременно иметь значение;
- атрибут uncertainty не должен иметь значение;
- типы неопределенности должны быть REAL.

Определения инвариантов на языке OCL:

```
inv "либо value, либо uncertainRange": not
    (value.oclIsDefined and uncertainRange.isNotNull)
inv "либо null, либо value": isNull xor (value.oclIsDefined or
    uncertainRange.isNotNull)
inv "типы неопределенности": uncertainRange.low.oclIsDefined
    implies uncertainRange.low.oclIsKindOf("REAL") and
      uncertainRange.high.oclIsDefined implies
      uncertainRange.high.oclIsKindOf("REAL") and
      uncertainty.oclIsDefined implies
      uncertainty.oclIsKindOf("REAL")
```

7.8.7.6 Операции

7.8.7.6.1 plus[+] (other : REAL) : REAL: значение суммы текущего значения (this) и другого значения (other).

7.8.7.6.2 minus[-] (other : REAL) : REAL: значение разности текущего значения (this) и другого значения (other).

7.8.7.6.3 times[*] (other : REAL) : REAL: значение произведения текущего значения (this) на другое значение (other).

7.8.7.6.4 negated[-] : REAL: изменение знака текущего значения (this).

7.8.7.6.5 dividedBy[/] (other : REAL) : REAL: значение деления текущего значения (this) на другое значение (other). Если другое значение равно 0, то результат имеет причину пустоты nullFalvor с кодом NI.

7.8.7.6.6 abs() : REAL: абсолютная величина текущего значения (this).

7.8.7.6.7 floor() : INT: наибольшее целое число, меньшее текущего значения (this) или равное ему.

7.8.7.6.8 ceiling() : INT: наименьшее целое число, большее текущего значения (this) или равное ему.

7.8.7.6.9 round() : INT: целое число, ближайшее к текущему значению (this). Если таких чисел два, берется наибольшее из них.

7.8.7.6.10 inverted() : REAL: значение 1, деленное на текущего значение (this).

7.8.7.6.11 max(other : REAL) : REAL: максимум из текущего значения (this) и другого значения (other).

7.8.7.6.12 min(other : REAL) : REAL: минимум из текущего значения (this) и другого значения (other).

7.8.7.6.13 power(other : REAL) : REAL: текущее значение (this), возведенное в степень, равную другому значению (other).

7.8.7.6.14 toInterval(): IVL(REAL): преобразование текущего значения (this) в интервал, отражающий точность этого значения.

7.8.7.6.15 comparable(other : QTY) : BL: вещественные числа всегда можно сравнить.

7.8.7.7 Примеры

7.8.7.7.1 Точность

```
<example xsi:type="REAL" value="23.0005"/>
```

Число с плавающей точкой 23.0005.

```
<example xsi:type="REAL" value="23.00"/>
```

Число с плавающей точкой 23.00.

7.8.7.7.2 Неопределенность

```
<example xsi:type="REAL" value="23" uncertaintyType="N">
    <uncertainty xsi:type="REAL" value="0.87"/>
</example>
```

Число с плавающей точкой 23. Известно, что его неопределенность имеет нормальное распределение со стандартным отклонением 0.87. Обратите внимание, что для атрибута uncertainty всегда надо задавать тип данных с помощью конструкции xsi:type.

7.8.8 Тип данных RTO (отношение)

7.8.8.1 Описание

Специализация типа данных QTY

Количество, представленное как отношение числителя к знаменателю.

Общие множители в числителе и знаменателе автоматически не сокращаются.

Тип данных RTO используется для титров (например, «1:128») и других величин в результатах лабораторных анализов, которые действительно представляют собой отношения. Отношения не являются просто «структурированными числовыми данными», поэтому измерения артериального давления (например, «120/60») не являются отношениями.

Примечания

1 Отношения отличаются от рациональных чисел тем, что в них общие множители числителя и знаменателя никогда не сокращаются. Отношение двух вещественных или целых чисел автоматически не понижается до вещественного числа. Этот тип данных определен не для общего представления рациональных чисел. Он используется только в том случае, когда общие множители числителя и знаменателя не предполагается сокращать. Такое происходит не часто. При передаче результатов измерений отношения встречаются почти исключительно в титрах. Во многих других случаях вместо типа данных RTO можно использовать тип данных REAL.

2 Поскольку во многих технологиях реализации предполагается, что параметризованные типы данных являются коллекциями или имеют только один параметр, то в настоящем стандарте тип данных RTO не представлен как параметризованный. Ограничения, вводимые при применении типа данных RTO, определят, какая в нем используется форма типа данных QTY.

7.8.8.2 Синтаксис ИСО/МЭК 11404

```
type RTO = class (
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    expression : ED,
    originalText : ED.TEXT,
    uncertainty : QTY,
    uncertaintyType : UncertaintyType,
```

```
    uncertainRange : IVL(QTY)
    numerator : QTY,
    denominator : QTY
)
```

7.8.8.3 Атрибуты

7.8.8.3.1 numerator : QTY: числитель, то есть величина, представляющая собой делимое в отношении.

7.8.8.3.2 denominator : QTY: знаменатель, то есть величина, представляющая собой делитель в отношении.

7.8.8.3.3 Делитель не должен равняться нулю.

7.8.8.4 Равенство

Два непустых значения типа RTO равны, если попарно равны их числители и знаменатели.

7.8.8.5 Инварианты:

- если значение типа RTO не имеет причины пустоты nullFlavor, то оба атрибута numerator и denominator требуются;
- атрибут неопределенности (uncertainty) не должен иметь значение;
- ни атрибут numerator, ни атрибут denominator не могут иметь тип данных TS.

Представление инвариантов на языке OCL:

```
inv "атрибуты numerator и denominator требуются": isNull xor
    (numerator.isNotNull and denominator.isNotNull)
inv "у атрибутов типа данных RTO нет истории или режима изменения":
    noUpdateOrHistory(numerator) and
      noUpdateOrHistory(denominator)
inv "атрибут uncertainty отсутствует": uncertainty.oclIsUndefined and
    uncertainRange.oclIsUndefined
inv "тип данных TS запрещен": (numerator.isNotNull implies not
    numerator.oclIsKindOf("TS")) and (denominator.isNotNull
    implies not denominator.oclIsKindOf("TS"))
```

7.8.8.6 Операции

7.8.8.6.1 comparable(other : QTY) : BL: текущее значение (this) и другое значение (other) могут быть сравнимыми, если сравнимы и их числители, и их знаменатели.

7.8.8.7 Примеры

```
<example xsi:type="RTO">
    <numerator xsi:type="MO" value="103.00" currency="USD"/>
    <denominator xsi:type="PQ" value="1" unit="day"/>
</example>
```

103 доллара США в день.

Следует учесть, что внутренние декларации xsi:type требуются всегда.

7.8.9 PQ (физическая величина)

7.8.9.1 Описание

Специализация типа данных QTY

Размерностная величина, представляющая результат измерения.

7.8.9.2 Синтаксис ISO 11404

```
type PQ = class (
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
```

```
    flavorId : Set(characterstring),
    expression : ED,
    originalText : ED.TEXT,
    uncertainty : QTY,
    uncertaintyType : UncertaintyType,
    uncertainRange : IVL(QTY)
    value : Decimal,
    codingRationale : CodingRationale,
    unit : characterstring,
    translation : Set(PQR)
)
```

7.8.9.3 Атрибуты

7.8.9.3.1 value : Decimal: произведение числа на единицу измерения, представляющее значение типа PQ или PQR, не имеющее причины пустоты nullFlavor.

7.8.9.3.2 unit : Code: единица измерения, указанная согласно системе унифицированных кодов единиц измерения UCUM (Unified Code for Units of Measure).

В системе UCUM определены две формы выражений: чувствительная к регистру и нечувствительная к регистру. В представлении значений типа PQ используются коды, чувствительные к регистру. Форме, чувствительной к регистру, присвоен ОИД 2.16.840.1.113883.6.8. По умолчанию единице измерения присвоен код UCUM со значением «1» (единица).

Для равенства физических величин не требуется, чтобы у них соответственно совпадали свойства value и unit. Эти свойства относятся только к способу представления физических величин. К примеру, 1 м равен 100 см. Хотя различны и единицы, и количества, эти физические величины равны. Таким образом, надо не рассчитывать на то, что физическая величина имеет конкретную единицу измерения, а обеспечивать автоматическое преобразование разных сопоставимых единиц.

Единицы должны быть взяты из системы UCUM, обеспечивающей однозначное представление единиц измерений. Не всегда бывает очевидно, на какую единицу измерения должны быть отображены некоторые измерения, осуществляемые в сфере здравоохранения.

Общий шаблон результата измерения имеет вид «количество» (value) «единиц» (unit) «предмета». В этой схеме тип данных PQ представляет количество и единицы, а «предмет» задается некоторым кодированным понятием, связанным со значением типа PQ контекстом использования. Некоторые результаты измерений представляются в этой форме вполне очевидно, например, «Температура тела пациента» равна «37» «градусов Цельсия», или «250» «мг/сут» «салицилата».

Однако для некоторых результатов измерений, осуществляемых в сфере здравоохранения, такое представление не столь очевидно. Двумя классическим примерами служат «5 бутылок пива» и «3 таблетки ацетаминофена». Проблема в том, что в системе UCUM отсутствуют такие единицы, как «бутылка», «таблетка» или «стаканы».

Это обусловлено тем, что ни таблетки, ни шарики не являются правильными единицами. Что за таблетки? Насколько велик стакан? В такого рода случаях для обеспечения интероперабельности понятие, которое кажется единицей, должно быть раскрыто далее. Если требуется правильно указать количество, то обычно следует указание точного значения результата измерения с соответствующей единицей, определенной в системе UCUM. Если это невозможно, то понятие не является частью результата измерения. Для подобных случаев в системе UCUM предусмотрена единица измерения, называемая «единицей» (unity). Правильным способом представления таких результатов измерения служит «3» «1» «таблетки» ацетаминофена, где «1» — код «единицы» в системе UCUM, а «предмет» имеет квалификатор. Для получения дополнительной квалифицирующей информации необходим контекст использования.

7.8.9.3.3 codingRationale : CodingRationale: причина, по которой предоставляется значение типа PQ или PQR. Может быть указано несколько причин. Допустимые значения см. в 7.5.1.4.10.

7.8.9.3.4 translation : Set(PQR): перевод, то есть альтернативное представление той же самой физической величины в других единицах из другой системы кодирования единиц и, возможно, с другим значением количества.

Элементы обработки информации не обязаны проверять, что перевод является правильным преобразованием основной единицы, но они могут это делать и отклонять экземпляры типа данных, у которых такие переводы не являются правильными.

Примечание — Переводы могут содержать другие представления физической величины в терминах единиц системы UCUM, но обычно это не практикуется, поскольку в этой системе одну форму единиц можно привести к другой.

7.8.9.4 Равенство

Два экземпляра типа REAL равны, если атрибуты value и units в их канонических формах равны, либо их интервалы неопределенности uncertainRange не пусты, не имеют причины пустоты nullFlavor и равны. Атрибуты codingRationale, source и translation в проверке на равенство не участвуют.

7.8.9.5 Инварианты:

- если экземпляр типа REAL не имеет причины пустоты nullFlavor, то атрибуты value или uncertainRange должны иметь значение;
- атрибуты value и uncertainRange не могут одновременно иметь значение;
- атрибуты uncertainty должны иметь тип данных PQ;
- если атрибуты uncertainty указаны, их канонические формы должны совпадать.

Определение инвариантов на языке OCL:

```
def: let unitsMatch (other : PQ) : Boolean =
    (canonical.unit = other.canonical.unit)

inv "либо null, либо value": isNull xor (value.oclIsDefined or
    uncertainRange.isNotNull)
inv "value xor uncertainRange": not (value.oclIsDefined and
    uncertainRange.isNotNull)
inv "uncertain types": uncertainRange.low.oclIsDefined implies
    uncertainRange.low.oclIsKindOf(PQ) and
    uncertainRange.high.oclIsDefined implies
      uncertainRange.high.oclIsKindOf(PQ) and
      uncertainty.oclIsDefined implies
            uncertainty.oclIsKindOf(PQ)
inv "неопределенности - канонические формы":
    uncertainRange.low.oclIsDefined implies
      unitsMatch(uncertainRange.low) and
      uncertainRange.high.oclIsDefined implies
            unitsMatch(uncertainRange.high) and
            uncertainty.oclIsDefined implies unitsMatch(uncertainty)

context PQ::isDifference(other : QTY): Boolean
  post: other.oclIsKindOf(PQ) and
    canonical.unit.equals(other.oclAsTypeOf(PQ).canonical.unit)
```

7.8.9.6 Операции

7.8.9.6.1 canonical : PQ: значение, приведенное к каноническим единицам измерения. В описании системы UCUM приведена подробная информация о канонических единицах.

7.8.9.6.2 comparable(other : QTY) : BL: текущее значение (this) и другое значение (other) могут сравниваться, если одинаковы единицы в их канонических формах.

7.8.9.6.3 inverted() : PQ: Обратная величина для значения типа PQ. Должны быть обращены значения обоих атрибутов value и unit.

7.8.9.6.4 negated[-] : PQ: изменение знака текущего значения (this).

7.8.9.6.5 plus[+] (other : PQ) : PQ: значение суммы текущего значения (this) и другого значения (other); если единицы измерения не совпадают, то причина пустоты nullFlavor с кодом NI.

7.8.9.6.6 minus[-] (other : PQ) : PQ: значение разности текущего значения (this) и другого значения (other); если единицы измерения не совпадают, то причина пустоты nullFlavor с кодом NI.

7.8.9.6.7 times[*] (other : PQ) : PQ: значение произведения текущего значения (this) на другое значение (other) с соответствующими изменениями единиц измерения.

7.8.9.6.8 times[*] (other : REAL) : PQ: значение произведения текущего значения (this) на другое значение (other).

7.8.9.6.9 dividedBy[/] (other : PQ) : PQ: значение деления текущего значения (this) на другое значение (other) с соответствующими изменениями единиц измерения. Если другое значение равно 0, то результат имеет причину пустоты nullFalvor с кодом NI.

7.8.9.6.10 dividedBy[/] (other : REAL) : PQ: значение деления текущего значения (this) на другое значение (other). Если другое значение равно 0, то результат имеет причину пустоты nullFalvor с кодом NI.

7.8.9.6.11 abs() : PQ: абсолютная величина текущего значения (this).

7.8.9.6.12 max(other : PQ) : PQ: максимум из текущего значения (this) и другого значения (other); если единицы измерения не совпадают, то причина пустоты nullFlavor с кодом NI.

7.8.9.6.13 min(other : PQ) : PQ: минимум из текущего значения (this) и другого значения (other); если единицы измерения не совпадают, то причина пустоты nullFlavor с кодом NI.

7.8.9.6.14 toInterval() : IVL(PQ): преобразование текущего значения (this) в интервал, отражающий точность этого значения.

7.8.9.7 Примеры

7.8.9.7.1 Простое значение

```
<example xsi:type="PQ" value="1.1" unit="mg/mL"/>
```

1.0 мг/мл.

```
<example xsi:type="PQ" value="1.1" unit="mg/mL" codingRationale="R">
    <translation codingRationale="O" value="0.0011"
      codeSystem="2.16.840.1.113883.6.8" code="grams/litre"/>
</example>
```

1.0 мг/мл как перевод исходного результата измерения 0.0011 грамм/литр, представленного в местной системе кодирования, в единицы системы UCUM.

7.8.9.7.2 Диапазоны неопределенности

```
<doseQuantity xsi:type="PQ" units="1">
    <uncertainRange>
      <low xsi:type="PQ" value="1"/>
      <high xsi:type="PQ" value="2"/>
    </uncertainRange>
</doseQuantity>
```

Это значение типа URG(PQ) указывает, что пациент должен принимать 1—2 таблетки. Оно может быть частью рецепта, гласящего «при наличии сильной боли принимать перорально 1—2 таблетки каждые 4—6 часов, но не более 8 в день». Следует обратите внимание на то, что атрибут имеет значение «1», то есть единица в системе UCUM по умолчанию, поэтому она совпадает с единицами границ low и high атрибута интервала неопределенности uncertainRange.

7.8.9.7.3 Использование выражений

```
<substanceAdministration>
    ...
    <doseQuantity xsi:type="PQ" nullFlavor="DER" unit="mL">
      <expression mediaType="application/hl7-factor+xml">
            <xml>
                  <coefficient value="30" unit="mL/kg"/>
                  <factor value="bodyMass"/>
            </xml>
      </expression>
    </doseQuantity>
    ...
    <derivedFrom>
      <localVariableName value="bodyMass"/>
      <monitoringObservation>
            <code code="29463-7" codeSystem="2.16.840.1.113883.21..6492">
                  <displayName value="BODY WEIGHT:MASS:PT:^PATIENT:QN"/>
```

```
        </code>
      </monitoringObservation>
    </derivedFrom>
</substanceAdministration>
```

В этом примере демонстрируется использование в атрибуте expression специального языка, разработанного рабочей группой HL7.

Дозировка субстанции, принимаемой пациентом, зависит от массы его тела, а именно 30 мг на килограмм массы тела. Поскольку эта масса может быть неизвестной или измениться, то вместо указания конкретного значения, рассчитанного по массе тела, задается максимальная величина дозы в терминах выражения. Так как атрибут value не указан, должна присутствовать причина пустоты nullFlavor; ее значение DER является вполне адекватным выбором при использовании выражения expression.

Значение массы тела может быть получено из результата исследования с кодом 29463-7 в классификации LOINC; если такие исследования не найдены, то результат применения выражения, указанного в атрибуте expression, должен быть пустым или иметь причину пустоты nullFlavor, поскольку величина дозы не может быть рассчитана.

Примечание — Язык Factor документирован в спецификации HL7 Abstract Data Types.

7.8.10 Тонкость PQ.TIME

7.8.10.1 Описание

Ограничение типа данных PQ

Тонкость PQ.TIME ограничивает тип данных PQ единицами, описывающими период времени.

7.8.10.2 Инварианты:

- единицы измерения должны быть единицами времени [например, «s» (секунда), «min» (минута), «h» (час), «d» (день), «wk» (неделя), «a» (год)].

Определение инвариантов на языке OCL:

```
inv "должны быть единицами времени": canonical.unit = "s"
```

7.8.11 Тонкость PQR (Physical Quantity Representation)

7.8.11.1 Описание

Специализации тонкости CD.CV

Расширение типа данных кодированных значений, представляющее физическую величину с указанием единиц измерения, взятых из некоторой системы кодирования. Используется для альтернативного представления физической величины. Кодированное значение указывает единицу измерения (обычно взятую из некоторой системы кодирования, отличающейся от системы UCUM).

7.8.11.2 Синтаксис ИСО/МЭК 11404

```
type PQR = class (
  validTimeLow : characterstring,
  validTimeHigh : characterstring,
  controlInformationRoot : characterstring,
  controlInformationExtension : characterstring,
  nullFlavor : NullFlavor,
  updateMode : UpdateMode,
  flavorId : Set(characterstring),
  code : characterstring,
  codeSystem : characterstring,
  codeSystemName : characterstring,
  codeSystemVersion : characterstring,
  valueSet : characterstring,
  valueSetVersion : characterstring,
  displayName : ST,
  originalText : ED.TEXT,
  codingRationale : CodingRationale),
  translation : Set(CD),
```

```
    source : CD,
    value : Decimal
)
```

7.8.11.3 Атрибуты

7.8.11.3.1 Value : Decimal: количество измеренного значения в терминах единиц, указанных в атрибуте code.

7.8.11.4 Равенство

Два значения типа PQR равны, если поларно равны их атрибуты value, code и codeSystem. Другие атрибуты не участвуют в проверке на равенство.

7.8.11.5 Инварианты:

- атрибут unit требуется;
- атрибут source недопустим;
- атрибут originalText недопустим.

Примечание — Может быть указан лишь один экземпляр атрибута originalText, а именно тот, который относится к самой физической величине.

Представление инвариантов на языке OCL:

```
inv "либо null, либо value": isNull xor value.oclIsDefined
inv "атрибут originalText недопустим": originalText.oclIsUndefined
inv "у тонкости PQR нет истории и режима изменений": noUpdateOrHistory
inv "переводы недопустимы": translation.isEmpty
inv "атрибут source недопустим": source.oclIsUndefined
```

7.8.12 Тип данных MO (денежная сумма)

7.8.12.1 Описание

Специализация типа данных QTY

Тип данных MO описывает величину, выражающую денежную сумму в некоторой валюте.

Валюты — единицы, в которых указаны денежные суммы в разных экономических регионах. В то время как денежная сумма является единственным видом количества (денег), обменные курсы валют являются переменными. Это принципиальное отличие между физическим количеством PQ и денежными количествами MO, а также причина того, почему единицы валюты не являются единицами физических величин.

7.8.12.2 Синтаксис ИСО/МЭК 11404

```
type MO = class (
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    expression : ED,
    originalText : ED.TEXT,
    uncertainty : QTY,
    uncertaintyType : UncertaintyType,
    uncertainRange : IVL(QTY)
    value : Decimal,
    currency : characterstring
)
```

7.8.12.3 Атрибуты

7.8.12.3.1 value : Decimal: величина денежной суммы. Значения типа MO обычно имеют точность до 0.01 (один цент, пенс, пайса и т. д.) или 1 (йена, форинт и т. д.), однако бывают и другие степени точности. В ИСО 4217 указана точность суммы для большинства валют.

7.8.12.3.2 currency : Code: код денежной единицы в соответствии с ИСО ISO 4217.

7.8.12.4 Равенство

Два значения типа MO равны, если попарно равны их атрибуты value и currency, либо их интервалы неопределенности uncertainRange не пусты, не имеют причин пустоты nullFlavor и равны.

7.8.12.5 Инварианты:

- если экземпляр типа MO не имеет причины пустоты nullFlavor, то атрибуты value или uncertainRange должны иметь значение;
- если экземпляр типа MO не имеет причины пустоты nullFlavor, то атрибут currency должен быть определен;
- атрибуты value и uncertainRange не могут одновременно иметь значение;
- атрибуты uncertainty должны иметь тип данных PQ;
- если атрибуты uncertainty указаны, их канонические единицы должны совпадать.

Представление инвариантов на языке OCL:

```
def: let currencyMatches (other : MO) : Boolean =
    currency = other.currency

inv "либо null, либо currency": isNull xor currency.oclIsDefined
inv "либо null, либо value": isNull xor (value.oclIsDefined or
    uncertainRange.isNotNull)
inv "либо value, либо uncertainRange": not (value.oclIsDefined and
    uncertainRange.isNotNull)
inv "типы неопределенности": uncertainRange.low.oclIsDefined implies
    uncertainRange.low.oclIsKindOf(MO) and
    uncertainRange.high.oclIsDefined implies
      uncertainRange.high.oclIsKindOf(MO) and
      uncertainty.oclIsDefined implies
            uncertainty.oclIsKindOf(MO)
inv "неопределенности - валюты":
    uncertainRange.low.oclIsDefined implies
      currencyMatches(uncertainRange.low) and
      uncertainRange.high.oclIsDefined implies
            currencyMatches(uncertainRange.high) and
            uncertainty.oclIsDefined implies
                  currencyMatches(uncertainty)

context CO::isDifference(other : QTY): Boolean
post: other.oclIsKindOf(MO)
```

7.8.12.6 Операции

7.8.12.6.1 plus[+] (other : MO) : MO: значение суммы текущего значения (this) и другого значения (other); если единицы измерения не совпадают, то причина пустоты nullFlavor с кодом NI.

7.8.12.6.2 minus[-] (other : MO) : MO: значение разности текущего значения (this) и другого значения (other); если единицы измерения не совпадают, то причина пустоты nullFlavor с кодом NI.

7.8.12.6.3 times[*] (other : REAL) : MO: значение произведения текущего значения (this) на другое значение (other).

7.8.12.6.4 dividedBy[/] (other : REAL) . MO: значение деления текущего значения (this) на другое значение (other). Если другое значение равно 0, то результат имеет причину пустоты nullFalvor с кодом NI.

7.8.12.6.5 max(other : MO) : MO: максимум из текущего значения (this) и другого значения (other).

7.8.12.6.6 min(other : MO) . MO: минимум из текущего значения (this) и другого значения (other).

7.8.12.6.7 comparable(other : QTY) : BL: текущее значение (this) и другое значение (other) могут сравниваться, если одинаковы их валюты.

7.8.12.7 Примеры

```
<example xsi:type="MO" value="42" currency="AUD"/>
```

42 австралийских доллара.

7.8.13 Тип данных TS (момент времени)

7.8.13.1 Описание

Специализация типа данных QTY

Определение: величина, указывающая момент на оси естественного времени. Момент времени чаще всего представляют как календарное выражение.

7.8.13.2 Синтаксис ИСО/МЭК 11404

```
type TS - class {
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    expression : ED,
    originalText : ED.TEXT,
    uncertainty : QTY,
    uncertaintyType : UncertaintyType,
    uncertainRange : IVL(QTY)
    value : characterstring
}
```

7.8.13.3 Атрибуты

7.8.13.3.1 value : String: значение типа данных TS. Атрибут value представляет собой строку в формате «ГГГГ[ММ[ДД[ЧЧ[ММ[СС[.С[С[С[С]]]]]]]][+/-ППпп», соответствующую ограничению стандарта ИСО 8601, определенному в разделе 32 (всемирное время) стандарта ИСО 8824 (ASN.1). Этот формат должен использоваться с необходимой степенью точности.

7.8.13.4 Равенство

Два непустых значения типа данных TS равны только в том случае, если попарно равны их атрибуты time и precision либо их интервалы неопределенности uncertainRange не пусты, не имеют причин пустоты и равны. Если у обоих значений типа TS указаны часовые пояса, то перед сравнением их значения должны быть приведены к одному часовому поясу. Если часовой пояс указан только у одного значения типа TS, то результатом сравнения является признак пустоты с кодом NI.

7.8.13.5 Инварианты:

- если значение типа TS не имеет причины пустоты nullFlavor, то атрибуты value или uncertainRange должны присутствовать;
- если атрибут value присутствует, должен быть указан по меньшей мере полный год;
- атрибуты uncertainty должны иметь тип данных PQ с единицами времени;
- атрибуты value и uncertainRange не могут одновременно иметь значение.

Представление инвариантов на языке OCL:

```
def: let hasTimezone : Boolean - value.pos("+") > 0
    or value.pos(" ") > 0
inv "либо value,либо uncertainRange": not (value.oclIsDefined and
        uncertainRange.isNotNull)
inv "либо null,либо value": isNotNull implies (value.oclIsDefined
        or uncertainRange.isNotNull)
inv "типы неопределенности": uncertainRange.low.oclIsDefined implies
       uncertainRange.low.oclIsKindOf(PQ) and
      uncertainRange.high.oclIsDefined implies
       uncertainRange.high.oclIsKindOf(PQ) and
      uncertainty.oclIsDefined implies
       uncertainty.oclIsKindOf(PQ)
inv "неопределенности - единицы": uncertainRange.low.oclIsDefined
      implies uncertainRange.canonical.unit - "s" and
```

```
      uncertainRange.high.oclIsDefined implies
      uncertainRange.high.canonical.unit = "s" and
      uncertainty.oclIsDefined implies
      uncertainty.canonical.unit = "s"

context TS::isDifference(other : QTY): Boolean
  post: other.oclIsKindOf(TS) and canonical.unit = "s"
```

7.8.13.6 Операции

7.8.13.6.1 plus(+) (other : PQ) : TS: значение суммы текущего значения (this) и другого значения (other); если единицы измерения другого значения не являются единицами времени, то причина пустоты nullFlavor с кодом NI.

7.8.13.6.2 minus[-] (other : PQ) : TS: значение разности текущего значения (this) и другого значения (other); если единицы измерения другого значения не являются единицами времени, то причина пустоты nullFlavor с кодом NI.

7.8.13.6.3 minus[-] (other : TS) : PQ: значение разности текущего значения (this) и другого значения (other); возвращаемое значение должно выражаться в единицах времени.

7.8.13.6.4 max(other : TS) : TS: максимум из текущего значения (this) и другого значения (other).

7.8.13.6.5 min(other : TS) : TS: минимум из текущего значения (this) и другого значения (other).

7.8.13.6.6 toInterval() : IVL(TS): преобразование текущего значения (this) в интервал, отражающий точность этого значения..

7.8.13.6.7 precision() : Integer: число значащих цифр в штампе даты и времени.

7.8.13.6.8 comparable(other . QTY) : BL: текущее значение (this) и другое значение (other) можно сравнить, если другое значение имеет тип TS.

7.8.13.7 Примеры

7.8.13.7.1 Момент времени

```
<example xsi:type="TS" value="200311012345.1 0500"/>
```

23:45pm 1 ноября 2003 года в часовом поясе UTC-5 (например, US Eastern Standard Time).

7.8.13.7.2 Дата рождения

```
<example xsi:type="TS" value="1945"/>
```

Пациент родился в 1945 году. День и месяц не известны.

Результат преобразования даты этого примера в интервал (toIVL()) имеет следующий вид:

```
<example xsi:type="IVL_TS" lowClosed="true" highClosed="false">
    <low value="19450101000.0000"/>
    <high value="19460101000.0000"/>
</example>
```

7.8.14 Тонкость TS.DATE

7.8.14.1 Описание

Ограничение типа данных TS

Тонкость TS.DATE ограничивает тип данных TS только значениями дат.

7.8.14.2 Инварианты:

- часовой пояс недопустим;
- часы, минуты, секунды или миллисекунды недопустимы.

Определение инвариантов на языке OCL:

```
inv "дата": not hasTimezone and value.size <= 8
```

7.8.15 Тонкость TS.DATE.FULL

7.8.15.1 Описание

Ограничение тонкости TS.DATE

Тонкость TS.DATE.FULL ограничивает тонкость TS.DATE таким образом, чтобы в дате был указан конкретный день.

7.8.15.2 Инварианты:

- полное указание дня.

Определение инвариантов на языке OCL:

```
inv "полное указание дня": value.size = 8
```

7.8.16 Тонкость TS.DATETIME

7.8.16.1 Описание

Ограничение типа данных TS

Тонкость TS.DATETIME ограничивает тип данных TS с точностью до секунд.

7.8.16.2 Инварианты:

- миллисекунды должны иметь пустое значение.

Определения инвариантов на языке OCL:

```
inv "DateTime": (value.size <= 14) or (hasTimezone and
    value.size <= 19)
```

7.8.17 Тонкость TS.DATETIME.FULL

7.8.17.1 Описание

Ограничение тонкости TS.DATETIME

Тонкость TS.DATETIME.FULL ограничивает тонкость TS.DATETIME таким образом, чтобы была указана конкретная секунда в часовом поясе.

7.8.17.2 Инварианты:

- часовой пояс требуется;
- полное указание даты и времени, включая секунды и часовой пояс, требуется.

Определения инвариантов на языке OCL:

```
inv "полные дата и время": value.size = 19 and hasTimezone
```

7.8.18 Тонкость TS.INSTANT

7.8.18.1 Описание

Ограничение типа данных TS

Тонкость TS.INSTANT ограничивает тип данных TS таким образом, чтобы он содержал указание конкретного момента времени, включая доли секунды с точностью до четырех знаков и часовой пояс.

7.8.18.2 Инварианты:

- часовой пояс требуется;
- полное указание даты и времени, включая доли секунды с точностью до четырех знаков и часовой пояс, требуется.

Определения инвариантов на языке OCL:

```
inv "момент времени": value.size = 24 and hasTimezone
```

7.9 Коллекции типов данных

7.9.1 Введение

Эти типы данных являются коллекциями дискретных элементов. Они представлены на рисунке 8.

7.9.2 Тип данных COLL

7.9.2.1 Описание

Абстрактный; специализация типа данных ANY

Параметр: T : ANY

Коллекция значений, которые могут быть перечислены с помощью итератора.

7.9.2.2 Операции

7.9.2.2.1 size() : INT: Число элементов в коллекции.

7.9.2.2.2 includes(object : T) : BL: имеет значение true, если объект object является элементом коллекции. Для установления, действительно ли данный объект является элементом данной коллекции,

используется операция equals. Употребляется также термин «содержит»: коллекция содержит объект «o», если includes(o) возвращает значение true.

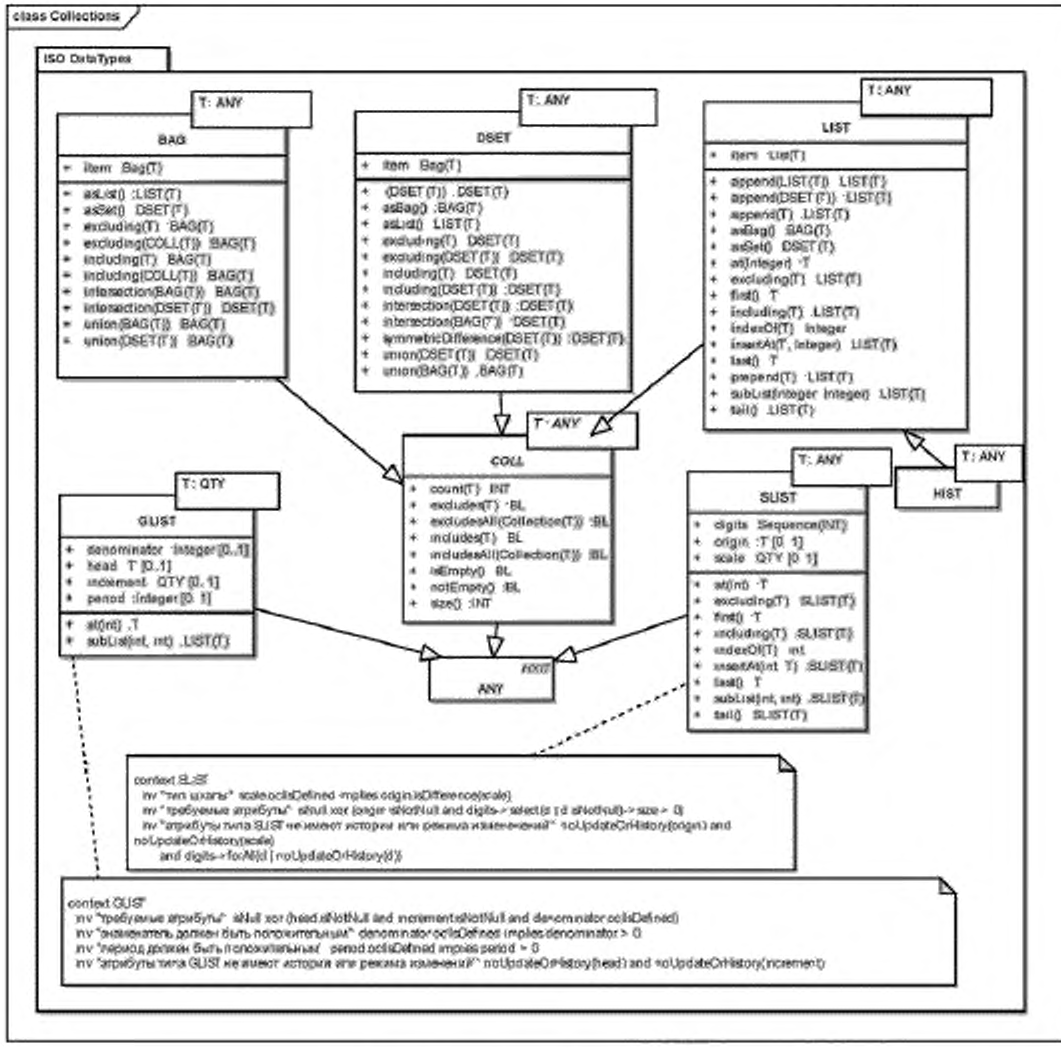


Рисунок 8 — Коллекции типов данных

7.9.2.2.3 excludes(object : T) : BL: имеет значение true, если объект object не является элементом коллекции. Для установления, действительно ли данный объект не является элементом данной коллекции, используется операция equals.

7.9.2.2.4 count(object : T) : INT: число экземпляров данного объекта в коллекции. Для установления, сколько раз данный объект входит в коллекцию, используется операция equals.

7.9.2.2.5 includesAll(c2 : Collection(T)) : BL: имеет значение true, если данная коллекция содержит все элементы коллекции c2.

7.9.2.2.6 excludesAll(c2 : Collection(T)) : BL: имеет значение true, если данная коллекция не содержит ни одного элемента коллекции c2.

7.9.2.2.7 isEmpty() : BL: имеет значение true, если данная коллекция пуста.

7.9.2.2.8 notEmpty() : BL: имеет значение true, если данная коллекция не пуста.

7.9.2.3 Равенство

Поскольку тип данных COLL является абстрактным, равенство его экземпляров не определено.

7.9.3 Тип данных DSET (дискретное множество)

7.9.3.1 Описание

Специализация типа данных COLL

Параметр: T : ANY

Неупорядоченная коллекция, содержащая различные дискретные значения.

Правильные (не имеющие причин пустоты nullFlavor) дискретные множества не должны содержать дублирующие элементы. В контексте использования должно быть определено, каким образом сравниваются элементы при проверке их уникальности во множестве. По умолчанию определение уникальности основано на правилах равенства, определенных в настоящем стандарте. Дискретные множества не должны содержать равные значения и не должны содержать пустые элементы или элементы, имеющие причину пустоты nullFlavor, для которых равенство не может быть определено. При фактическом использовании дискретного множества в контексте его использования может быть указано альтернативное определение уникальности элементов. Оно может разрешать наличие в правильном множестве элементов с причиной пустоты nullFlavor. Во избежание неоднозначности в объявлениях соответствия элементов обработки информации, предусматривающих альтернативные определения уникальности элементов множества, должно быть точно указано, как такие определения представлены.

В то время как правильные множества (не имеющие элементов с причинами пустоты nullFlavor) не содержат элементы, не отвечающие определению уникальности, дискретные множества с причинами пустоты nullFlavor могут содержать элементы с дублируемыми значениями или со значениями, имеющими причины пустоты nullFlavor. Учтите, что дискретные множества, помеченные как обязательные, не могут иметь причину пустоты nullFlavor и поэтому не могут содержать такие значения.

7.9.3.2 Синтаксис ИСО/МЭК 11404

```
type DSET (T : ANY) = class (
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    item : Bag(T)
)
```

Примечание — Тип данных DSET описывает множество дискретных элементов. Если элементы принадлежат упорядоченному, но дискретному домену (например, целые числа типа INT) и представляют собой последовательность смежных значений, например, 2, 3, 4, 5, то это же самое множество может быть описано типом данных QSET, к примеру, IVL(2..5). Однако во всех других случаях между экземплярами типов данных QSET и DSET нет пересечений.

7.9.3.3 Атрибуты

7.9.3.3.1 item : Bag(T): содержание множества

Это пример шаблона обертки примитивного типа данных. Дополнительные сведения приведены в 6.3.

Элементы содержатся в мультимножестве, поскольку контекст использования точно определяет, как задается уникальность элементов экземпляра типа DSET. Понятие множества в ядре языка OCL предполагает фиксированное определение равенства, а именно равенства, задаваемого для типа данных, которое может быть менее детальным, чем в контексте использования. Хотя внутренняя структура позволяет дубликаты, все элементы множества должны быть уникальными в соответствии с определением, предоставленным контекстом использования. Если в контексте это определение не выделено, то по умолчанию используются определения равенства, приведенные в настоящем стандарте.

Примечание — Формальное определение ограничений уникальности здесь не приводится, поскольку оно является достаточно сложным и его польза не очевидна.

7.9.3.4 Равенство

Два непустых экземпляра типа данных DSET равны, если они содержат те же самые элементы.

Примечания

1 Определение содержания элемента основано на той же семантике равенства, которая описана в настоящем стандарте, поэтому вполне возможно, к примеру, что экземпляр типа DSET(CD) может быть равен экземпляру типа DSET(CS).

2 Экземпляр типа DSET(INT) может быть равен экземпляру типа QSET(INT), если они содержат одни и те же элементы.

7.9.3.5 Инварианты:

Отсутствуют.

Примечание — Правила об истории или режиме изменений применимы к элементу, где использован тип данных BAG(T).

7.9.3.6 Операции

7.9.3.6.1 union(s : DSET(T)) : DSET(T): множество, содержащее все элементы текущего множества (this) и элементы другого множества (s) после удаления дубликатов. Операция union идентична операции including.

7.9.3.6.2 union(bag : BAG(T)) : BAG(T): мультимножество, содержащее все элементы текущего множества (this) и элементы другого мультимножества (bag). Дубликаты удаляются.

7.9.3.6.3 intersection(s : DSET(T)) : DSET(T): пересечение текущего множества (this) и другого множества (s), то есть все элементы, содержащиеся одновременно и в this, и в s.

7.9.3.6.4 intersection(bag : BAG(T)) : DSET(T): пересечение текущего множества (this) и мультимножества bag.

7.9.3.6.5 minus[–](s : DSET(T)) . DSET(T): элементы текущего множества (this), не входящие во множество s.

7.9.3.6.6 including(object : T) : DSET(T): множество, содержащее все элементы текущего множества (this), а также элемент object, если он еще не входит в текущее множество. Операция including идентична операции union.

7.9.3.6.7 including(other : DSET(T)) : DSET(T): множество, содержащее все элементы текущего множества (this) и элементы другого множества (other) после удаления дубликатов.

7.9.3.6.8 excluding(object : T) : DSET(T): множество, содержащее все элементы текущего множества (this) за вычетом элемента object.

7.9.3.6.9 excluding(other : DSET(T)) : DSET(T): множество, содержащее все элементы текущего множества (this) за вычетом элементов другого множества (other).

7.9.3.6.10 symmetricDifference(s : DSET(T)) : DSET(T): множество, содержащее все элементы, которые входят либо в текущее множество (this), либо во множество s.

7.9.3.6.11 asList() : LIST(T): последовательность, содержащая все элементы текущего множества (this) в произвольном порядке.

7.9.3.6.12 asBag() : BAG(T): мультимножество, содержащее все элементы текущего множества (this).

7.9.3.7 Примеры

7.9.3.7.1 Целочисленные множества

```
<example xsi:type="DSET_INT">
    <item value="3"/>
    <item value="6"/>
    <item value="9"/>
    <item value="11"/>
</example>
```

Множество целых чисел (3, 6, 9, 11). Следует обратить внимание на то, что тип множества содержит указание типа элементов в этом множестве (указание xsi:type="DSET" некорректно, поскольку тип данных DSET является параметрическим).

```
<example xsi:type="DSET_INT">
<item value="11"/>
```

```
<item value="6"/>
<item value="9"/>
<item value="3"/>
</example>
```

Множество, описанное в этом примере, идентично предыдущему — порядок элементов не имеет во множествах никакого значения.

7.9.3.7.2 Проблемы со множествами

```
example xsi:type="DSET_TEL">
    <item value="tel:+15536667777" use="H"/>
    <item nullFlavor="UNK" use="WP"/>
</example>
```

Множество телефонных номеров, в которое входит известный номер домашнего телефона и неизвестный номер служебного телефона. Важно учесть, что это множество не является правильным. Множества не могут содержать пустые значения.

```
<example xsi:type="DSET_TEL" nullFlavor="UNK">
  <item value="tel:+15556667777" use="H"/>
  <item nullFlavor="UNK" use="WP"/>
</example>
```

То же самое множество, но представленное правильным образом. Поскольку одно из значений неизвестно, множество в целом также неизвестно.

Собственно, по этой причине адреса и номера телефонов не должны моделироваться как множества типа DSET, поскольку в том случае, когда они помечены как обязательные, неизвестные номера уже не могут быть в них представлены.

```
<example xsi:type="DSET_REAL">
    <item value="3.1"/>
    <item value="3.5"/>
</example>
```

Если тип элементов не является дискретным (PQ, MO, REAL), то дискретное множество довольно неоднозначно. Например, является число 3.103 элементом этого множеств или нет? В связи с этим типы данных DSET не должны использоваться с параметрами, имеющими эти типы.

7.9.4 Тип данных LIST (последовательность)

7.9.4.1 Описание

Специализация типа данных COLL

Параметр: T : ANY

Коллекция, содержащая дискретные (но не обязательно разные) элементы в определенной последовательности. Этим элементам присвоено смещение от начала списка; первый элемент имеет смещение 0.

Последовательность является упорядоченной коллекцией элементов, но назначение порядка не вытекает непосредственно из последовательности. Это назначение должно быть определено при использовании последовательности. Следует учесть, что в некоторых случаях порядок фиксирован (например, в типе данных HIST), но в других случаях он не является фиксированным и указан только смысл упорядочения (например, в типах данных EN, AD).

7.9.4.2 Синтаксис ИСО/МЭК 11404

```
type LIST (T : ANY) = class (
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
```

```
    flavorId : Set(characterstring),
    item : Sequence(T)
)
```

7.9.4.3 Атрибуты

7.9.4.3.1 item : Sequence(T): содержание последовательности

Это пример шаблона обертки примитивного типа данных. Дополнительные сведения приведены в 6.3.

7.9.4.4 Равенство

Два списка равны, если они оба пусты либо содержат те же самые элементы в том же самом порядке.

Примечания

1 Определение содержания элемента основано на семантике равенства, описанной в настоящем стандарте, так что, к примеру, экземпляр списка LIST(ED) может быть равен экземпляру списка LIST(ST).

2 Равенство экземпляров типа SLIST основан на равенстве последовательности значений, поэтому экземпляр списка SLIST<T> может быть равен экземпляру списка LIST<T>.

7.9.4.5 Инварианты:

Отсутствуют.

Примечание — Правила об истории или режиме изменений применимы к элементу, где используется BAG(T).

7.9.4.6 Операции

7.9.4.6.1 append(s : LIST(T)) : LIST(T): последовательность типа LIST, состоящая из всех элементов текущей последовательности (this), за которыми следуют все элементы последовательности s.

7.9.4.6.2 append(s : DSET(T)) . LIST(T): последовательность типа LIST, состоящая из всех элементов текущей последовательности (this), за которыми следуют все элементы множества s в произвольном порядке. Эта операция идентична операции including.

7.9.4.6.3 append (object: T) : LIST(T): последовательность типа LIST, состоящая из всех элементов текущей последовательности (this), за которыми следует элемент object.

7.9.4.6.4 prepend(object : T) : LIST(T): последовательность типа LIST, состоящая из элемента object, за которым следуют все элементы текущей последовательности (this).

7.9.4.6.5 insertAt(object : T, index : Integer) : LIST(T): последовательность типа LIST, образованная из текущей последовательности (this), в которую вставлен элемент object в позиции index. Если эта позиция больше длины текущей последовательности или равна этой длине, то результатом операции будет пустая последовательность.

7.9.4.6.6 subList(lower : Integer, upper . Integer) : LIST(T): фрагмент текущей последовательности (this), образованный элементами, находящимися в позициях с lower по upper. Если позиции lower или upper больше длины текущей последовательности либо равны этой длине, либо lower или upper меньше 0, или lower больше upper, то результатом операции будет пустая последовательность.

7.9.4.6.7 at(i : Integer) : T: элемент текущей последовательности, находящийся в позиции i. Если позиция i больше длины текущей последовательности либо равна этой длине, то результатом операции будет пустое значение. Первый элемент последовательности имеет позицию i = 0.

7.9.4.6.8 indexOf(obj : T) : Integer: позиция элемента obj в текущей последовательности или пустое значение, если число вхождений этого элемента в последовательность отлично от 1 (вообще не входит либо входит несколько раз).

7.9.4.6.9 first() : T: первый элемент текущей последовательности (this) или пустое значение, если последовательность пуста.

7.9.4.6.10 last() : T: последний элемент текущей последовательности (this) или пустое значение, если последовательность пуста.

7.9.4.6.11 tail() : LIST(T): текущая последовательность, из которой удален первый элемент, или пустое значение, если последовательность пуста.

7.9.4.6.12 including(object : T) : LIST(T): последовательность типа LIST, состоящая из всех элементов текущей последовательности (this), за которыми следует элемент object. Эта операция идентична операции append.

7.9.4.6.13 excluding(object : T) : LIST(T): последовательность типа LIST, состоящая из всех элементов текущей последовательности (this) за исключением всех экземпляров элемента object. Порядок оставшихся элементов не изменяется.

7.9.4.6.14 asBag() : BAG(T): мультимножество, состоящее из всех элементов текущей последовательности (this), включая дубликаты, или признак пустоты nullFlavor.

7.9.4.6.15 asSet() : DSET(T): множество, состоящее из всех элементов текущей последовательности (this), из которых удалены дубликаты.

7.9.4.7 Примеры

```
<example xsi:type="LIST_INT">
  <item value="3"/>
  <item value="11"/>
  <item value="6"/>
  <item value="9"/>
</example>
```

Целочисленная последовательность. Порядок элементов существенен и всегда должен соблюдаться.

7.9.5 Тип данных GLIST (генерируемая последовательность)

7.9.5.1 Описание

Специализация типа данных ANY.

Параметр: T : QTY

Периодическая или монотонная последовательность значений, не перечисляемых, а генерируемых по небольшому числу параметров. Используется для задания регулярных моментов считывания биосигналов.

7.9.5.2 Синтаксис ИСО/МЭК 11404

```
type GLIST (T : ANY) = class (
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    head: T,
    increment : QTY,
    denominator : integer,
    period : integer
)
```

7.9.5.3 Атрибуты

7.9.5.3.1 head : T: первый элемент последовательности. Это начальный элемент генерируемой последовательности.

7.9.5.3.2 increment : QTY: приращение, то есть разность между одним элементом и предшествующим ему отличающимся элементом.

Пример — При генерировании последовательности (1; 4; 7; 10; 13; ...) приращение increment равно 3; аналогичным образом при генерировании последовательности (1; 1; 4; 4; 7; 7; 10; 10; 13; 13; ...) приращение increment также равно 3. Фактический тип данных QTY определяется параметром T. Значение приращения должно быть положительным.

7.9.5.3.3 denominator : Integer: целое число, на которое делится позиция элемента последовательности. Задает число повторений того же самого значения элемента последовательности, перед тем как перейти к следующему значению.

Пример — При генерировании последовательности (1; 1; 1; 2; 2; 2; 3; 3; 3; ...) знаменатель denominator равен 3.

Свойство denominator применяется для генерирования нескольких последовательностей, используемых для периодического сканирования многомерного пространства. Например, (абстрактный) экран

телевизора использует два таких генератора для строк и столбцов изображения. Скажем, если на экране отображается развертка 200 строк с 320 растровыми столбцами, то у генератора столбцов свойство denominator равно 1, а у генератора строк оно равно 320.

7.9.5.3.4 period : Integer: если это свойство не пусто и не имеет причины пустоты nullFlavor, то оно определяет чередование последовательности, то есть через заданное в нем число различных элементов значение элемента последовательности возвращается к начальному значению.

Пример — Последовательность (1; 2; 3; 1; 2; 3; 1; 2; 3; ...) имеет период 3; последовательность (1; 1; 2; 2; 3; 3; 1; 1; 2; 2; 3; 3; ...) также имеет период 3.

Задание свойства period позволяет периодически повторять группу значений. «Сигнал» такого периодического генератора всегда является «пилой» наподобие изображения линейной функции на осциллографе.

7.9.5.4 Равенство

Тип данных GLIST описывает генератор последовательностей. Два экземпляра типа GLIST равны, если они задают одну и ту же последовательность элементов.

Следует учесть, что последовательности, генерируемые с помощью экземпляров типа GLIST, бесконечны, а экземпляр типа LIST не может быть бесконечным, поэтому они никогда не могут быть равны.

7.9.5.5 Инварианты:

- если экземпляр типа GLIST не имеет причины пустотыt nullFlavor, то требуются все атрибуты, кроме period;
- свойство denominator должно быть положительным;
- период должен быть положительным.

Определения инвариантов на языке OCL:

```
inv "требуемые атрибуты": isNull xor (head.isNotNull and
    increment.isNotNull and denominator.oclIsDefined)
inv "значение атрибута denominator должно быть положительным":
    denominator.oclIsDefined
      implies denominator > 0
inv "период должен быть положительным": period.oclIsDefined implies
    period > 0
inv "у атрибута значения типа GLIST нет истории или режима изменений":
    noUpdateOrHistory(head) and noUpdateOrHistory(increment)
```

7.9.5.6 Операции

7.9.5.6.1 subList(lower : Integer, upper : Integer) : LIST(T): фрагмент последовательности, определяемой данным экземпляром типа GLIST, начиная с позициит lower и завершая позицией upper (включительно). Если позиции lower или upper больше длины текущей последовательности либо равны этой длине, либо lower или upper меньше 0, или lower больше upper, то результатом операции будет пустая последовательность.

7.9.5.6.2 at(i : Integer) : T: элемент текущей последовательности, находящийся в позиции i. Если позиция i больше длины текущей последовательности либо равна этой длине, то результатом операции будет пустое значение.

7.9.5.7 Примеры

```
<example xsi:type="GLIST_PQ" period="100" denominator="100">
    <head value="0" unit="V"/>
    <increment xsi:type="PQ" value="1" unit="mV"/>
</example>
```

Считывание линейно возрастающего напряжения на входе цифрового осциллографа в пределах от 0 до 100 мВ за 100 шагов по 1 мВ. Частота опроса по этим данным не определяется, поскольку интервал времени между последовательными считываниями неизвестен.

```
<example xsi:type="GLIST_TS" denominator="1">
    <head value="20020729203000"/>
```

```
    <increment xsi:type="PQ" value="100" unit="us"/>
</example>
```

Временной ряд, значения которого начинаются в 20:30 29 июня 2002 года и возрастают на 100 мкс при каждом шаге. Если скомбинировать его с предыдущим генератором в качестве моментов считывания сигнала, то напряжение будет меняться на 1 мВ каждые 100 мкс. Поскольку период составляет 100 шагов, то его длительность равна 10 мс, что эквивалентно частоте сигнала 100 Гц.

Другие примеры приведены в таблице 24.

Таблица 24 — Примеры генерируемых последовательностей

Свойство head	Свойство increment	Свойство denominator	Свойство period	Описание последовательности
0	1	1	NullFlavor.PINF	Последовательность идентификаторов, в которой каждый элемент равен своей позиции
198706051900	2 h	1	NullFlavor.PINF	Последовательность значений времени, начинающаяся с 19:00 5 июня 1987 года и каждый раз возрастающая на два часа: 21:00, 23:00, 1:00 (6 июня), 3:00, 5:00 и т. д.
0 V	1 mV	1	100	Считывание линейно возрастающего напряжения на входе цифрового осциллографа в пределах от 0 до 100 мВ за 100 шагов по 1 мВ. Частота опроса по этим данным не определяется, поскольку интервал времени между последовательными считываниями неизвестен
2002072920300	100 us	1	NullFlavor.PINF	Временной ряд, значения которого начинаются в 20:30 29 июня 2002 года и возрастают на 100 мкс при каждом шаге. Если скомбинировать его с предыдущим генератором в качестве моментов считывания сигнала, то напряжение будет меняться на 1 мВ каждые 100 мкс. Поскольку период составляет 100 шагов, то его длительность равна 10 мс, что эквивалентно частоте сигнала 100 Гц
0 V	1 mV	100	100	Сочетание этого генератора с двумя предыдущими может описывать трехмерное пространство считываний с двумя осями напряжения и осью времени. Каждое приращение напряжения данного генератора также составляет 100 мВ с периодичностью 100 шагов, однако изменение напряжения происходит только через 100 считываний, поэтому первый генератор напряжения успевает сделать полный цикл за одно приращение напряжения в данном генераторе. Эти два генератора напряжения можно представить как «строки» и «столбцы» считываемого «кадра». Если в качестве таймера используется предыдущий генератор, то частота кадров 100 мВ · 100 мВ составит 1 Гц

7.9.6 Тип данных SLIST (последовательность считываний)

7.9.6.1 Описание

Специализация типа данных ANY

Параметр: T : QTY

Последовательность считываемых величин, получаемая из списка целых значений с помощью масштабирования и сдвига. Используется для описания считываемых биосигналов.

7.9.6.2 Синтаксис ИСО/МЭК 11404

```
type SLIST (T : ANY) = class {
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
```

```
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    origin : T,
    scale : QTY,
    digits : Sequence(INT)
)
```

7.9.6.3 Атрибуты

7.9.6.3.1 origin : T: начало шкалы значений элементов последовательности, то есть физическая величина, которой соответствует нулевое значение элемента последовательности.

7.9.6.3.2 scale : QTY: величина масштаба, на которую умножаются значения элементов последовательности. Фактический тип данных QTY определяется параметром T.

7.9.6.3.3 digits : Sequence(INT): последовательность целых значений считываний. Обычно это исходные значения, получаемые от аналого-цифрового преобразователя.

7.9.6.4 Равенство

Тип данных SLIST является специализацией типа данных LIST. Два экземпляра типа SLIST равны, если они задают одну и ту же последовательность значений.

Примечание — Поскольку тип данных SLIST является специализацией типа данных LIST и равенство основано на свойствах типа данных LIST, то экземпляр типа данных LIST<PQ> может быть равен экземпляру типа данных SLIST<PQ>.

7.9.6.5 Инварианты:

- масштаб scale должен иметь тип разности значений, имеющих тип начала шкалы origin;
- если экземпляр типа SLIST не имеет причины пустоты nullFlavor, то атрибут origin и по меньшей мери один атрибут digit требуются.

Определения инвариантов на языке OCL:

```
  inv "требуемые атрибуты": isNull xor (origin.isNotNull and
      digits->select(d   d.isNotNull)->size > 0)
  inv "тип масштаба": scale.oclIsDefined implies
 origin.isDifference(scale)
  inv "no updateMode or History on SLIST attributes":
     noUpdateOrHistory(origin) and noUpdateOrHistory(scale)
       and digits >forAll(d | noUpdateOrHistory(d))
```

7.9.6.6 Операции

7.9.6.6.1 insertAt(object : T, index : Integer) : SLIST(T): последовательность типа SLIST, образованная из текущей последовательности (this) типа SLIST, в которую вставлен элемент object в позиции index. Если эта позиция больше длины текущей последовательности или равна этой длине, то результатом операции будет пустая последовательность.

7.9.6.6.2 subList(lower : Integer, upper : Integer) : SLIST(T): фрагмент текущей последовательности (this), образованный элементами, находящимися в позициях с lower по upper. Если позиции lower или upper больше длины текущей последовательности либо равны этой длине, либо lower или upper меньше 0, или lower больше upper, то результатом операции будет пустая последовательность.

7.9.6.6.3 at(i : Integer) : T: элемент текущей последовательности, находящийся в позиции i. Если позиция i больше длины текущей последовательности либо равна этой длине, то результатом операции будет пустое значение.

7.9.6.6.4 indexOf(obj : T) : Integer: позиция элемента obj в текущей последовательности или пустое значение, если число вхождений этого элемента в последовательность отлично от 1.

7.9.6.6.5 first() : T: первый элемент текущей последовательности (this) или пустое значение, если последовательность пуста.

7.9.6.6.6 last() : T: последний элемент текущей последовательности (this) или пустое значение, если последовательность пуста.

7.9.6.6.7 tail() : SLIST(T): текущая последовательность, из которой удален первый элемент, или пустое значение, если последовательность пуста.

7.9.6.6.8 including(object : T) : SLIST(T): последовательность типа SLIST, состоящая из всех элементов текущей последовательности (this), за которыми следует элемент object.

7.9.6.6.9 excluding(object : T) : SLIST(T): последовательность типа SLIST, состоящая из всех элементов текущей последовательности (this) за исключением всех экземпляров элемента object. Порядок оставшихся элементов не изменяется.

7.9.6.7 Примеры

```
<example xsi:type="SLIST_PQ">
    <origin value='0' unit='uV'/>
    <scale xsi:type="PQ" value='2.5' unit='uV'/>
    <digit value="-4"/>
    <digit value=" 13"/>
    <digit value="-18"/>
    <digit value=" 18"/>
    <digit value=" 18"/>
    <digit value="-17"/>
    <digit value=" 16"/>
    <digit value="-16"/>
    <digit value=" 16"/>
    <digit value=" 16"/>
    <digit value=" 16"/>
    <digit value=" 17"/>
    <digit value=" 18"/>
    <digit value=" 18"/>
    <digit value=" 1"/>
    <digit value=" 17"/>
    <digit value=" 16"/>
    <digit value=" 16"/>
    <digit value="-16"/>
    <digit value="--15"/>
    <digit value="-13"/>
    <digit value=" 11"/>
    <digit value="-10"/>
    <digit value="-10"/>
    <digit value=" 9"/>
    <digit value="--6"/>
    <digit value=" 4"/>
    <digit value=" 5"/>
    <digit value=" 5"/>
    <digit value=" 3"/>
    <digit value=" 2"/>
    <digit value="-2"/>
    <digit value=" 1"/>
    <digit value="1"/>
    <digit value="2"/>
    <digit value="3"/>
    <digit value=""/>
    <digit value="7"/>
    <digit value="8"/>
    <digit value="9"/>
    <digit value="10"/>
    <digit value="11"/>
    <digit value="12"/>
    <digit value="13"/>
```

```
    <digit value="15"/>
    <digit value="17"/>
    <digit value="19"/>
    <digit value="21"/>
    <digit value="23"/>
    <digit value="25"/>
    <digit value="27"/>
    <digit value="29"/>
    <digit value="30"/>
    <digit value="30"/>
    <digit value="31"/>
    <digit value="34"/>
    <digit value="37"/>
    <digit value="40"/>
    <digit value="43"/>
    <digit value="45"/>
    <digit value="4"/>
    <digit value="46"/>
    <digit value="46"/>
    <digit value="46"/>
    <digit value="46"/>
    <digit value="47"/>
    <digit value="49"/>
    <digit value="51"/>
    <digit value="53"/>
    <digit value="55"/>
    <digit value="57"/>
    <digit value="59"/>
    <digit value="60"/>
    <digit value="59"/>
    <digit value="58"/>
    <digit value="58"/>
    <digit value="58"/>
    <digit value="57"/>
    <digit value="56"/>
    <digit value="56"/>
    <digit value="56"/>
    <digit value="57"/>
    <digit value="57"/>
    <digit value="5"/>
    <digit value="53"/>
    <digit value="50"/>
    <digit value="47"/>
    <digit value="45"/>
    <digit value="74"/>
    <digit value="51"/>
    <digit value="38"/>
    <digit value="33"/>
    <digit value="31"/>
    <digit value="28"/>
    <digit value="25"/>
    <digit value="21"/>
    <digit value="16"/>
    <digit value="14"/>
    <digit value="15"/>
    <digit value="13"/>
```

```
    <digit value="9"/>
    <digit value="7"/>
    <digit value="4"/>
    <digit value="1"/>
    <digit value=" 1"/>
    <digit value=" 3"/>
    <digit value=" 4"/>
    <digit value=" 6"/>
    <digit value="·10"/>
    <digit value=" 12"/>
    <digit value=" 13"/>
    <digit value="·12"/>
    <digit value=" 12"/>
    <digit value=" 17"/>
    <digit value="·18"/>
    <digit value="·18"/>
    <digit value=" 18"/>
    <digit value="·19"/>
    <digit value=" 20"/>
    <digit value="·2."/>
    <digit value=" 20"/>
    <digit value=" 20"/>
    <digit value=" 20"/>
    <digit value=" 20"/>
    <digit value=" 2"/>
    ...
    <digit value="2"/>
    <digit value="1"/>
    <digit value="0"/>
    <digit value="0"/>
    <digit value="0"/>
    <digit value="1"/>
    <digit value="2"/>
    <digit value="2"/>
    <digit value="1"/>
    <digit value="1"/>
    <digit value="1"/>
    <digit value="0"/>
    <digit value="·1"/>
    <digit value="0"/>
    <digit value="1"/>
    <digit value="1"/>
    <digit value="1"/>
    <digit value="1"/>
    <digit value="2"/>
    <digit nullFlavor="UNK"/>
</example>
```

В этом примере показано отведение II электрокардиограммы, у которой начало отсчета калибровано величиной 0 мкВ, а масштаб равен 2,5 мкВ. Последнее измерение оказалось неудачным (чтобы показать пример значения с причиной пустоты nullFlavor).

7.9.7 Тип данных HIST (история)

7.9.7.1 Описание

Специализация типа данных LIST

Параметр: T : ANY

Коллекция, элементы которой исторически упорядочены.

7.9.7.2 Синтаксис ИСО/МЭК 11404

```
type HIST (T : ANY) = class (
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    item : Set(T)
)
```

Примечание — История относится скорее к правильности информации, нежели к знанию системами актуальной информации. Другие сведения приведены в 7.3.2.

7.9.7.3 Инварианты:

- все элементы последовательности должны иметь либо непустой атрибут validTimeLow, либо непустой атрибут validTimeHigh;
- периоды действительности, задаваемые этими атрибутами, не должны пересекаться, и элементы должны быть упорядочены в порядке возрастания хронологии.

Определения инвариантов на языке OCL:

```
inv "срок действия требуется ": item->forAll(i |
    i.validTimeLow.oclIsDefined or
    i.validTimeHigh.oclIsDefined)
```

7.9.7.4 Примеры

```
<example xsi:type="HIST_TEL">
  <item nullFlavor="UNK" use="WP H" validTimeHigh="199206"/>
  <item value="tel:+15552225543" use="H" validTimeLow="199206"
     validTimeHigh="199207"/>
  <item value="tel:+15556667777" use="H" validTimeLow="199207"/>
</example>
```

В этом примере показана известная история изменения номеров домашнего телефона.

7.9.8 Тип данных BAG (мультимножество)

7.9.8.1 Описание

Специализация типа данных COLL

Параметр: T : ANY

Неупорядоченная коллекция значений, каждое из которых может входить в эту коллекцию более одного раза.

7.9.8.2 Синтаксис ИСО/МЭК 11404

```
type BAG (T : ANY) = class (
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    item : Bag(T)
)
```

7.9.8.3 Атрибуты

7.9.8.3.1 item : Bag(T): содержание мультимножества

Это пример шаблона обертки примитивного типа данных. Дополнительные сведения приведены в 6.3.

7.9.8.4 Равенство

Два мультимножества равны, только если они оба пусты или оба содержат те же самые элементы, каждый из которых присутствует в них одинаковое число раз.

Примечание — Определение содержания элемента основано на той же семантике равенства, которая описана в настоящем стандарте, поэтому вполне возможно, к примеру, что экземпляр типа BAG(CD) может быть равен экземпляру типа BAG(CO).

7.9.8.5 Инварианты:

Отсутствуют.

Примечание — Правила об истории или режиме изменений применимы к элементу, где используется BAG(T).

7.9.8.6 Операции

7.9.8.6.1 union(bag : BAG(T)) . BAG(T): мультимножество, содержащее все элементы текущего мультимножества (this) и элементы другого мультимножества (bag). Если элемент содержится несколько раз, то в результате объединения мультимножеств общее число его экземпляров будет равно сумме его экземпляров в мультимножества this и bag. Операция union идентична операции including.

7.9.8.6.2 union(set : DSET(T)) : BAG(T): объединение текущего мультимножества (this) и множества set. Если элемент содержится и в мультимножестве this, и в множестве set, то в результате объединения общее число его экземпляров будет на единицу больше числа его экземпляров в мультимножестве this. Операция union идентична операции including.

7.9.8.6.3 intersection(bag : BAG(T)) : BAG(T): пересечение текущего мультимножества (this) и мультимножества bag.

7.9.8.6.4 intersection(set : DSET(T)) : DSET(T): пересечение текущего мультимножества (this) и множества set.

7.9.8.6.5 including(object : T) : BAG(T): мультимножество, содержащее все элементы текущего мультимножества (this) и элемент object. Если элемент object уже присутствует в мультимножестве this, то в результате данной операции число его вхождений увеличится на единицу.

7.9.8.6.6 including(coll: COLL(T)) : BAG(T): мультимножество, содержащее все элементы текущего мультимножества (this) и все элементы коллекции coll. Эта операция идентична операции union.

7.9.8.6.7 excluding(object : T) : BAG(T): мультимножество, содержащее все элементы текущего мультимножества (this), за исключением всех вхождений элемента object.

7.9.8.6.8 excluding(coll: COLL(T)) : BAG(T): мультимножество, содержащее все элементы текущего мультимножества (this), за исключением всех элементов коллекции coll.

7.9.8.6.9 asList() : LIST(T): последовательность, содержащая все элементы текущего мультимножества (this) в произвольном порядке.

7.9.8.6.10 asSet() : DSET(T): последовательность, содержащая все элементы текущего мультимножества (this) с удаленными дубликатами.

7.9.8.7 Примеры

```
<example xsi:type="BAG_TEL">
  <item value="tel:+15556667777" use="H"/>
  <item nullFlavor="UNK" use="WP"/>
</example>
```

Мультимножество телефонных номеров, содержащее известный номер домашнего телефона и неизвестный номер служебного телефона.

```
<example xsi:type="BAG_TEL">
  <item nullFlavor="UNK" use="WP"/>
  <item value="tel:+15556667777" use="H"/>
</example>
```

Это мультимножество не равно предыдущему мультимножеству телефонных номеров — хотя порядок вхождения элементов в мультимножество не имеет значения, номер служебного телефона имеет причину пустоты nullFlavor, поэтому равенство не может быть установлено.

7.10 Типы непрерывных множеств

7.10.1 Введение

Типы данных, приведенные на рисунке 9, используются для представления коллекций данных.

7.10.2 Тип данных QSET (непрерывное множество)

7.10.2.1 Описание

Абстрактный; специализация типа данных ANY

Параметр: T : QTY

Неупорядоченное множество различных значений, являющихся величинами.

Параметром типа данных QSET может быть любой упорядоченный тип данных вне зависимости от того, является он дискретным или непрерывным. Если параметр упорядочен только частично, то все элементы экземпляра типа QSET должны принадлежать полностью упорядоченному подмножеству частично упорядоченного типа данных (например, экземпляр типа PQ полностью упорядочен только в том случае, когда единицы измерения совместимы; каждое значение в экземпляре типа QSET(PQ) должно иметь одну и ту же каноническую единицу измерения).

Тип данных QSET является абстрактным. Его конкретные специализации представлены как дерево выражений, образованных с помощью сочетания оператора (QSI, QSD, QSU, QSP) и типа компонента (QSC, QSS и IVL, для типа данных TS еще типы компонентов PIVL и EIVL).

Экземпляры специализаций типа QSET не должны содержать в качестве элементов пустые значения или значения с причинами пустоты nullFlavor.

7.10.2.2 Синтаксис ИСО/МЭК 11404

```
type QSET (T : ANY) = class {
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    originalText : ED.TEXT
}
```

7.10.2.3 Атрибуты

7.10.2.3.1 originalText: ED.TEXT: исходный текст, на основе которого кодируется экземпляр типа QSET (если он служит источником этого экземпляра.

Исходный текст может использоваться в структурированном интерфейсе пользователя, чтобы описать, что видел пользователь на экранной форме в качестве представления величины, либо в ситуации, когда пользователь диктует или непосредственно вводит текст. Это тот текст, который введен или произнесен пользователем.

Допускает использовать потомки типа данных QSET для хранения только исходного текста. В этом случае исходный текст существует без правильного значения величины. Содержание атрибута originalText не заменяет правильного значения. Если фактическое содержание множества типа QSET не является правильным, то для этого множества должна быть указана причина пустоты nullFlavor вне зависимости от того, имеет атрибут originalText значение или нет.

Исходный текст должен представлять собой выдержку из релевантной информации, взятой из оригинальных источников, а не указатель на нее либо ее полное воспроизведение. Поэтому исходный текст должен быть представлен в неформатированном виде. В специфичных обстоятельствах, когда контекст использования точно описан, атрибут исходного текста originalText может быть ссылкой на некоторый другой текстовый артефакт, для которого однозначно определены способы разрешения ссылки.

Примечание — Детали связывания ссылки originalText.reference с различными артефактами медицинской информации (например, с документом и кодированным результатом) не входят в область применения настоящего стандарта и могут быть предписаны в спецификациях, использующих настоящий стандарт.

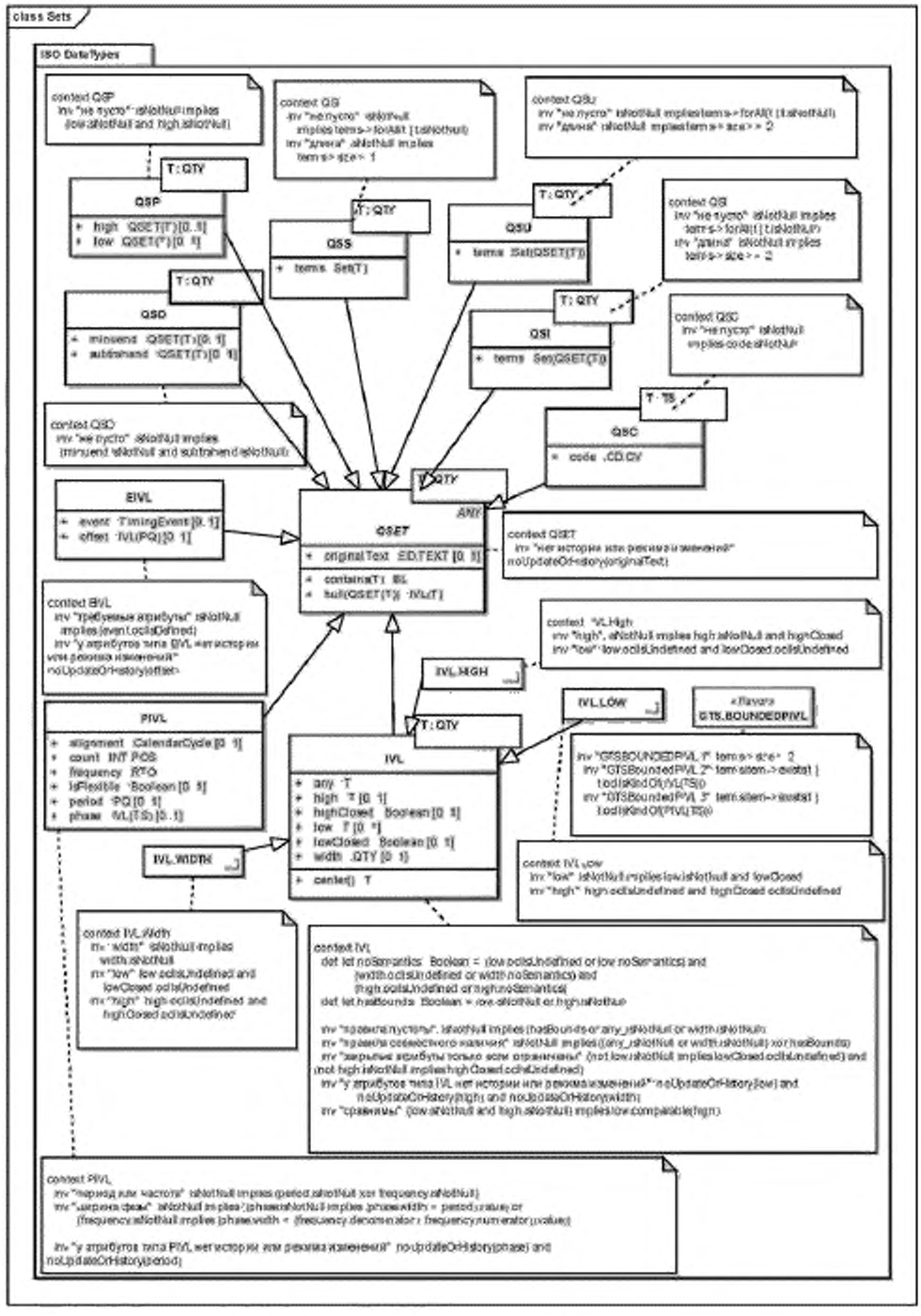


Рисунок 9 — Типы непрерывных множеств

7.10.2.4 Равенство

Отвлеченное определение равенства экземпляров типа данных QSET и всех его потомков, за исключением типа данных IVL, основано на принадлежности множеству: два экземпляра типа данных QSET равны, если они содержат одни и те же элементы. Однако тип данных QSET используется для построения деревьев выражений, которые могут становиться весьма сложными. Не всегда легко определить, описывают ли два различных дерева выражений типа QSET одно и то же множество элементов, поэтому определением равенства двух экземпляров типа данных QSET считается проверка на равенство, определенная для типа данных ANY по умолчанию.

Эта проверка на равенство применяется ко всем специализациям типа данных QSET, за исключением типа данных IVL, и не задана для других специализаций.

7.10.2.5 Инварианты

Представление инвариантов на языке OCL:

```
inv "нет истории или режима изменений ":
    noUpdateOrHistory(originalText)
```

7.10.2.6 Операции

7.10.2.6.1 contains (x. T) : BL: имеет значение true, если экземпляр множества QSET содержит значение x.

7.10.2.6.2 hull (x: QSET(T)) : IVL(T) : выпуклая оболочка текущего множества (this) и множества x, представляющая собой наименьший интервал, содержащий множества this и x.

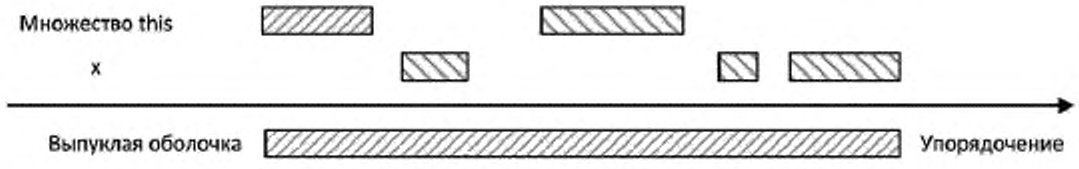


Рисунок 10 — Выпуклая оболочка

Примечание — Выпуклая оболочка экземпляра типа данных QSET может быть менее формально названа «связным внешним интервалом». Таким образом, выпуклая оболочка значения типа GTS описывает абсолютные начало и конец расписания. Для некоторых спецификаций множеств с бесконечными повторениями [например, описываемых типом данных PIVL(TS)] выпуклая оболочка имеет бесконечные границы. Термин «расписание» использован здесь в обобщенном смысле организованного ряда значений. Более привычное значение этого термина как плана событий, осуществляемых с течением времени, в точности описано типом данных QSET(TS).

7.10.2.7 Примеры

7.10.2.7.1 Тип данных QSET(TS)

Тип данных QSET(TS) известен также как общая спецификация времени GTS (General Timing Specification).

Первый пример указывает каждый второй вторник сезона отпусков (в США) с Дня поминовения до Дня труда в 2002—2003 годах. Он строится как пересечение следующих трех множеств:

- каждый второй вторник;
- 2002—2003 годы;
- сезон с Дня поминовения до Дня труда.

```
<example xsi:type="QSI_TS">
  <!-- пересечение, поскольку используется QSI -->

  <!-- каждый второй вторник -->
  <term xsi:type='PIVL_TS' alignment='DW'>
    <phase lowClosed='true' highClosed='false'>
      <low value='20001202'/>
      <high value='20001203'/>
```

```
    </phase>
    <period value='2' unit='wk'/>
  </term>

  <!-- 2002 - 2003 годы -->
  <term xsi:type='IVL_TS' lowClosed='true' highClosed='false'>
    <low value='20020101'/>
    <high value='20040101'/>
  </term>

  <!-- сезон с Дня поминовения до Дня труда -->
  <!-- периодическая оболочка от Дня поминовения до Дня труда -->
  <term xsi:type='QSP_TS'>
    <low xsi:type="QSI_TS">
      <!-- День поминовения: пересечение последней недели мая и понедельников
-->
      <term xsi:type='PIVL_TS'>
        <phase highClosed='false'>
          <low value='19870525'/>
          <high value='19870601'/>
        </phase>
        <period value='1' unit='a'/>
      </term>
      <term xsi:type='PIVL_TS'>
        <phase highClosed='false'>
          <low value='19870105'/>
          <high value='19870106'/>
        </phase>
        <period value='1' unit='wk'/>
      </term>
    </low>
    <high xsi:type="QSI_TS">
      <!-- День труда: пересечение первой недели сентября и понедельников -->
      <term xsi:type='PIVL_TS'>
        <phase highClosed='false'>
          <low value='19870901'/>
          <high value='19870908'/>
        </phase>
        <period value='1' unit='a'/>
      </term>
      <term xsi:type='PIVL_TS'>
        <phase highClosed='false'>
          <low value='19870105'/>
          <high value='19870106'/>
        </phase>
        <period value='1' unit='wk'/>
      </term>
    </high>
  </term>
</example>
```

7.10.3 Тип данных QSU (Объединение экземпляров типа данных QSET)

7.10.3.1 Описание

Специализация типа данных QSET.

Специализация типа данных QSET в форме объединения других множеств

7.10.3.2 Синтаксис ИСО/МЭК 11404

```
type QSU (T : ANY) = class (
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    originalText : ED.TEXT,
    terms : Set(QSET(T))
)
```

7.10.3.3 Атрибуты

7.10.3.3.1 terms : Set(QSET(T)): список экземпляров типа QSET, участвующих в объединении.

7.10.3.4 Инварианты:

- непустой экземпляр типа данных QSU может содержать только непустые экземпляры типа данных QSET;
- должны быть указаны как минимум два множества.

Определения инвариантов на языке OCL:

```
inv "не пусто": isNotNull implies terms->forAll(t |
    t.isNotNull)
inv "размер": isNotNull implies terms->size >= 2
```

7.10.4 Тип данных QSI (пересечение экземпляров типа данных QSET)

7.10.4.1 Описание

Специализация типа данных QSET

Специализация типа данных QSET в форме пересечения других множеств

7.10.4.2 Синтаксис ИСО/МЭК 11404

```
type QSI (T : ANY) = class (
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    originalText : ED.TEXT,
    terms : Set(QSET(T))
)
```

7.10.4.3 Атрибуты

7.10.4.3.1 terms : Set(QSET(T)): список экземпляров типа QSET, участвующих в пересечении.

7.10.4.4 Инварианты:

- непустой экземпляр типа данных QSI может быть результатом пересечения только непустых экземпляров типа данных QSET;
- должны быть указаны как минимум два множества.

Определения инвариантов на языке OCL:

```
inv "не пусто": isNotNull implies terms->forAll(t |
    t.isNotNull)
inv "размер": isNotNull implies terms->size >= 2
```

7.10.5 Тип данных QSD (разность экземпляров типа данных QSET)

Специализация типа данных QSET

Специализация типа данных QSET в форме разности двух множеств

7.10.5.1 Синтаксис ИСО/МЭК 11404

```
type QSU (T : ANY) = class {
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    originalText : ED.TEXT,
    minuend : QSET(T)
    subtrahend : QSET(T)
}
```

Разность представляет собой вычитание второго множества из первого.

7.10.5.2 Атрибуты

7.10.5.2.1 minuend : QSET(T): множество, из которого вычитается второе множество.

7.10.5.2.2 Subtrahend : QSET(T): множество, которое вычитается из первого множества.

7.10.5.3 Инварианты:

- непустой экземпляр типа данных QSD может быть результатом разности только непустых экземпляров типа данных QSET.

Определения инвариантов на языке OCL:

```
inv inv "не пусто": isNotNull implies (minuend.isNotNull and
    subtrahend.isNotNull)
```

7.10.6 Тип данных QSP (периодическая оболочка экземпляров типа данных QSET)

7.10.6.1 Описание

Специализация типа данных QSET

Специализация типа данных QSET в форме периодической оболочки двух множеств (см. рисунок 11).

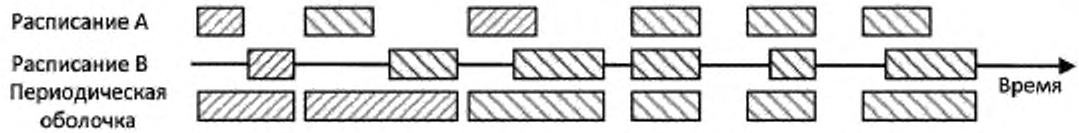


Рисунок 11 — Периодическая оболочка

Периодическая оболочка может быть получена с помощью сравнения двух чередующихся множеств. Значения A и B типа QSET считаются чередующимися, если интервалы событий обоих групп могут быть попарно упорядочены. При этом соответствующие пары интервалов a ⊆ A и b ⊆ B должны удовлетворять следующему условию: интервал a начинается до начала интервала b (или в то же самое время), и интервал b завершается после завершения интервала a (или в то же самое время).

Отношение чередования имеет место, если два расписания имеют одинаковую среднюю частоту и при этом второе расписание никогда не «перекрывает» первое расписание. Другими словами, никакой интервал события второго расписания не может начаться раньше соответствующего ему интервала события первого расписания.

7.10.6.2 Синтаксис ИСО/МЭК 11404

```
type QSP (T : ANY) = class {
    validTimeLow : characterstring,
```

```
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    originalText : ED.TEXT,
    low : QSET(T)
    high : QSET(T)
)
```

7.10.6.3 Атрибуты

7.10.6.3.1 low: QSET(T): множество, к которому применяется операция периодической оболочки.

7.10.6.3.2 High: QSET(T): множество, которое служит параметром операции периодической оболочки.

7.10.6.4 Инварианты:

- непустой экземпляр типа данных QSP может быть получен только из непустых экземпляров типа данных QSET.

Определения инвариантов на языке OCL:

```
inv inv "не пусто": isNotNull implies (low.isNotNull and
    high.isNotNull)
```

7.11.10 Тип данных QSS (перечисляемое множество типа QSET)

7.10.7.1 Описание

Специализация типа данных QSET

Специализация типа данных QSET в форме перечисления простых значений

Это сокращенная форма указания этих значений в форме отдельных интервалов.

7.10.7.2 Синтаксис ИСО/МЭК 11404

```
 type QSS (T : QTY) = class (
   validTimeLow : characterstring,
   validTimeHigh : characterstring,
   controlInformationRoot : characterstring,
   controlInformationExtension : characterstring,
   nullFlavor : NullFlavor,
   updateMode : UpdateMode,
   flavorId : Set(characterstring),
   originalText : ED.TEXT,
   terms : Set(T)
)
```

7.10.7.3 Атрибуты

7.10.7.3.1 terms : Set(T): совокупность значений, принадлежащих множеству. Это множество в действительности образуется как объединение интервалов, длина которых определяется точностью значений, подразумеваемой в определении типа данных T. Для некоторых подтипов типа данных QTY они будут либо вырожденными (тип данных INT) или неоднозначными (тип данных RTO), и тип данных QSS для таких подтипов будет лишен смысла. Тип данных QSS особенно полезен, если параметр имеет тип TS.

7.10.7.4 Инварианты:

- непустой экземпляр типа данных QSS может содержать только непустые значения;
- по крайней мере одно значение должно быть указано.

Определения инвариантов на языке OCL:

```
inv "не пусто": isNotNull implies terms->forAll(t |
     t.isNotNull)
inv "размер": isNotNull implies terms->size >= 1
```

7.10.7.5 Примеры

```
<example xsi:type='QSS_TS'>
   <term value='20071101'/>
   <term value='20071106'/>
</example>
```

Объединение интервалов времени, одним из которых является 1 ноября 2007 года, а другим — 6 ноября 2007 года.

7.10.8 Тип данных QSC (кодированное множество типа QSET)

7.10.8.1 Описание

Специализация типа данных QSET

Задает экземпляр типа данных QSET как кодированное значение, описывающее предопределенный экземпляр типа данных QSET(TS).

7.10.8.2 Синтаксис ИСО/МЭК 11404

```
type QSC (T : ANY) = class (
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    originalText : ED.TEXT,
    code : CD.CV
)
```

7.10.8.3 Атрибуты

7.10.8.3.1 code : CD.CV: предопределенный код, полностью и однозначно описывающий множество интервалов времени.

Возможный набор кодов, которые могут ,snm использоваys как значения данного атрибута, следует описать в объявлении соответствия стандарту. Рабочая группа HL7 определила набор кодов, составляющих систему кодирования GTSAbbrevation, и все элементы обработки информации, объявляющие непосредственное соответствие настоящему стандарту и поддерживающие тип данных QSC, должны поддерживать коды AM, PM, BID, TID, QID, JB и JE, принадлежащие этой системе (см. таблицу 25).

Таблица 25 — Фрагмент системы кодирования GTSAbbrevation, ОИД: 2.16.840.1.113883.5.1022 (требуемые коды, составляющие набор значений с ОИД 2.16.840.1.113883.1.11.10720)

Уровень	Код	Определение
1	AM	Каждое утро во время, назначенное конкретным учреждением
1	PM	Каждый день во время, назначенное конкретным учреждением
1	BID	Дважды в день во время, назначенное конкретным учреждением
1	TID	Трижды в день во время, назначенное конкретным учреждением
1	QID	Четырежды в день во время, назначенное конкретным учреждением
1	JB	Обычные рабочие дни (с понедельника по пятницу за вычетом праздников)
1	JE	Обычные выходные дни (суббота и воскресенье за вычетом праздников)

Наряду с этими кодами рабочая группа HL7 определила дополнительные коды, обозначающие некоторые официальные и неофициальные праздники в странах — источниках кодов (см. таблицу 26).

Таблица 26 — Фрагмент системы кодирования GTSAbbrevation, ОИД: 2.16.840.1.113883.5.1022 (дополнительные коды праздников, составляющие набор значений с ОИД 2.16.840.1.113883.1.11.10725)

Уровень	Код	Определение
1	JH	Праздники
2	JHCHR	Христианские праздники (римско/григорианская «западная» традиция)
3	JHCHRXME	Сочельник (24 декабря)
3	JHCHRXMS	Рождество (24 декабря)
3	JHCHRNEW	Новый год (1 января)
3	JHCHREAS	Пасхальное воскресенье. Дата Пасхи довольно сложно вычисляется по астрономическим таблицам, содержащим даты полнолуний. Детали вычислений можно найти на страницах gost.e-ecolog.ru и gost.e-ecolog.ru faq/docs/easter.html. Важно учесть, что православные христианские праздники основаны на юлианском календаре
3	JHCHRGFR	Страстная пятница, то есть пятница, непосредственно предшествующая Пасхальному воскресенью
3	JHCHRPEN	Пятидесятница, через семь недель после Пасхи (50-й день Пасхи)
2	JHNUS	Национальные праздники США (выходные дни федеральных служащих, установленные федеральным законом США 5 U.S.C. 6103.)
3	JHNUSMLK	День Мартина Лютера Кинга, третий понедельник января
3	JHNUSPRE	День рождения Вашингтона (День президентов), третий понедельник февраля
3	JHNUSMEM	День поминовения, последний понедельник мая
3	JHNUSMEM5	Пятница, непосредственно предшествующая Дню поминовения
3	JHNUSMEM6	Суббота, непосредственно предшествующая Дню поминовения
3	JHNUSIND	День независимости (4 июля)
3	JHNUSIND5	Альтернативная пятница перед выходными, предшествующими 4 июля [5 U.S.C. 6103(b)]
3	JHNUSIND1	Альтернативный понедельник после выходных, предшествующих 4 июля [5 U.S.C. 6103(b)]
3	JHNUSLBR	День труда, первый понедельник сентября
3	JHNUSCLM	День Колумба, второй понедельник октября
3	JHNUSVET	День ветеранов, 11 ноября
3	JHNUSTKS	День благодарения, четвертый четверг ноября
3	JHNUSTKS5	Пятница после Дня благодарения
2	JHNNL	Национальные праздники Нидерландов
3	JHNNLQD	День королевы (30 апреля)
3	JHNNLLD	День освобождения (5 мая каждые пять лет)
3	JHNNLSK	Синтерклаас (5 декабря)

Примечания

1 Эта таблица не полна и не содержит религиозные праздники, отличающиеся от традиционных («западных») григорианских, или национальные праздники других стран, кроме США и Нидерландов. Хотя другие страны могут использовать иную систему кодирования, они могут представить свои коды рабочей группе HL7 или организации ИСО для включения в данную систему кодирования.

2 Праздники имеют местную специфику. Какие именно религиозные праздники включены как потомки кода JH, зависит от местных и других традиций. Для обеспечения глобальной интероперабельности использование конструируемых выражений типа QSET безопаснее указания названий праздников. Однако некоторые праздники, которые зависят от лунных фаз (например, Пасха, Рамадан) или устанавливаются местным законодательством, не могут быть достаточно легко представлены как значения типа QSET, если не использовать тип данных QSC.

3 Элементы обработки информации могут определять собственный перечень кодов, создавая подходящий набор данных. При необходимости этот набор может быть использован в типе данных QSC.

7.10.8.4 Инварианты:

- атрибут code должен присутствовать.

Представление инвариантов на языке OCL:

```
inv "не пусто": isNotNull implies code.isNotNull
```

7.10.8.5 Примеры

```
<example xsi:type='QSC_TS' >
  <code code="JHCHRXMS" codeSystem="2.16.840.1.113883.5.1022"/>
</example>
```

Все дни Рождества.

7.10.9 Тип данных IVL (интервал)

7.10.9.1 Описание

Специализация типа данных QSET

Параметр: T : QTY

Множество последовательных значений упорядоченного типа данных.

В основу типа данных IVL может быть положен любой упорядоченный тип данных вне зависимости от того, является ли он дискретным или непрерывным. Если тип данных упорядочен только частично, то все элементы экземпляра типа данных IVL должны принадлежать полностью упорядоченному подмножеству частично упорядоченного типа данных. Например, тип данных PQ считается упорядоченным. Однако он упорядочен только частично; полное упорядочение определено только для сравнимых величин (имеющих одинаковую физическую размерность). Можно задать интервал между 2 и 4 м, но нельзя задать интервал между 2 м и 4 с.

7.10.9.2 Синтаксис ИСО/МЭК 11404

```
type IVL (T : ANY) = class (
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    originalText : ED.TEXT,
    low : T,
    lowClosed : boolean,
    high : T,
    highClosed : boolean,
    width : QTY,
    any : T
)
```

7.10.9.3 Атрибуты

7.10.9.3.1 low : T: нижняя граница интервала. Если она неизвестна, должна быть указана причина пустоты nullFlavor.

Нижняя граница не должна быть положительной бесконечностью.

7.10.9.3.2 lowClosed : Boolean: признак, включена нижняя граница в интервал (замкнутая граница) или исключена из него (открытая граница).

7.10.9.3.3 high : T: верхняя граница интервала. Если она неизвестна, должна быть указана причина пустоты nullFlavor.

Верхняя граница не должна быть отрицательной бесконечностью и должна быть не меньше нижней границы, если таковая существует.

7.10.9.3.4 highClosed : Boolean: признак, включена верхняя граница в интервал (замкнутая граница) или исключена из него (открытая граница).

7.10.9.3.5 width : QTY: разность между верхней и нижней границей. Атрибут width используется, если длина интервала известна, а точки начала и конца интервала неизвестны. Тип данных длины интервала (QTY) определяется типом параметра T.

7.10.9.3.6 any : T: указывает, что некоторое конкретное значение лежит в данном интервале.

Этот атрибут следует использовать, если неизвестно, когда что-то началось или закончилось, но известно, что оно произошло в определенное время. Эта часть имеет место для исследований (например, процессов течения заболевания), процедур и расписаний. В таких случаях ни нижняя, ни верхняя граница неизвестны, хотя может быть известна длина интервала.

7.10.9.4 Равенство

В отличие от других специализация типа данных QSET, равенство значений типа IVL определяется на основе принадлежности к множеству. Два значения типа IVL равны, если они содержать одни и те же элементы.

Примечания

1 При определении равенства значений типа IVL выделяются два особых случая. Если верхние границы пары значений являются положительной бесконечностью, то они равны; если нижние границы пары значений являются отрицательной бесконечностью, то они тоже равны.

2 Если два интервала имеют одну и ту же длину, но границы не известны, то они не считаются равными.

3 Это же правило применимо, если известно только то, что одно и то же значение (атрибута any) входит в оба интервала: такие интервалы никогда не считаются равными.

4 Поскольку равенство определяется принадлежностью к множеству, то экземпляры типов данных DSET(INT) и IVL(INT) могут быть равны. Например, множество 2, 3, 4 типа DSET(INT) равно интервалу 2..4 типа IVL(INT).

7.10.9.5 Инварианты:

- экземпляр типа данных IVL либо имеет причину пустоты nullFlavor, либо длину (width), либо принадлежащую ему точку (any), либо имеет нижнюю границу (low) и/или верхнюю границу (high). Если атрибуты any или width указаны, то атрибуты low и high должны отсутствовать. Если указаны атрибуты low или high, то должны отсутствовать атрибуты any и width;
- атрибуты lowClosed или highClosed могут быть указаны только в том случае, если указаны значения атрибутов low или high соответственно;
- значения атрибутов low и high должны быть сравнимы.

Представление инвариантов на языке OCL:

```
def: let hasBounds : Boolean = low.isNotNull or high.isNotNull
def: let noSemantics : Boolean = (low.oclIsUndefined or
     low.noSemantics) and (width.oclIsUndefined or
     width.noSemantics) and (high.oclIsUndefined or
     high.noSemantics)

inv "правила пустоты": isNonNull implies (hasBounds or any_.isNotNull
     or width.isNotNull)
inv "правила совместного присутствия ": isNotNull implies ((any_.isNotNull
     or width.isNotNull) xor hasBounds)
inv "граница может быть закрытой только в том случае, если она ограничена ":
    (not low.isNotNull implies lowClosed.oclIsUndefined) and
    (not high.isNotNull implies highClosed.oclIsUndefined)
inv "у атрибутов типа данных IVL нет истории и режима изменений ":
     noUpdateOrHistory(low) and
     noUpdateOrHistory(high) and noUpdateOrHistory(width)
inv "сравнимы": (low.isNotNull and high.IsNotNull) implies
     low.comparable(high)
```

7.10.9.6 Примеры

7.10.9.6.1 Целочисленный интервал

```
<example xsi:type='IVL_INT'>
   <low value='2'/>
   <high value='4'/>
</example>
```

Простой интервал значений типа INT между 2 и 4. Это то же самое, что указать множество (2, 3, 4) типа DSET.

Интервал значений типа PQ между 2,8 м включительно и 4,6 м исключительно.

7.10.9.6.2 Дневник операции

```
<example xsi:type='IVL_TS'>
   <width xsi:type='PQ' value='2' unit='h'/>
   <any value='200012C41000'/>
</example>
```

Операция длилась два часа и в момент времени 10:00 4 декабря 2000 года все еще продолжалась. Спецификация xsi:type значения длины width необходима, поскольку ее тип данных (QTY) является абстрактным.

7.10.10 Тонкость IVL.LOW

7.10.10.1 Описание

Ограничение типа данных IVL

Тонкость IVL.LOW ограничивает тип данных IVL таким образом, чтобы его атрибут нижней границы low присутствовал, а атрибут замыкания lowClosed имел значение true. Все остальные свойства запрещены.

7.10.10.2 Инварианты:

- атрибуты low и lowClosed должны иметь значения;
- атрибуты high и highClosed должны быть пустыми.

Определения инвариантов на языке OCL:

```
inv "нижняя граница ": isNotNull implies low.isNotNull and lowClosed
inv "верхняя граница": high.oclIsUndefined and highClosed.oclIsUndefined
```

7.10.11 Тонкость IIVL.HIGH

7.10.11.1 Описание

Ограничение типа данных IVL

Тонкость IVL.HIGH ограничивает тип данных IVL таким образом, чтобы его атрибут верхней границы high присутствовал, а атрибут замыкания highClosed имел значение true. Все остальные свойства запрещены.

7.10.11.2 Инварианты:

- атрибуты low и lowClosed должны быть пустыми;
- атрибуты high и highClosed должны иметь значения.

Определения инвариантов на языке OCL:

```
inv "верхняя граница": isNotNull implies high.isNotNull and highClosed
inv "нижняя граница": low.oclIsUndefined and lowClosed.oclIsUndefined
```

7.10.12 Тонкость IIVL.WIDTH

7.10.12.1 Описание

Ограничение типа данных IVL

Тонкость IVL.HIGH ограничивает тип данных IVL таким образом, что его атрибут длины width обязателен, а атрибуты low, lowClosed, high, highClosed запрещены.

7.10.12.2 Инварианты:

- атрибут width должен иметь значение;

- атрибуты low и lowClosed должны быть пустыми;
- атрибуты high и highClosed должны быть пустыми.

Определения инвариантов на языке OCL:

```
inv "длина": isNotNull implies width.isNotNull
inv "нижняя граница": low.oclIsUndefined and lowClosed.oclIsUndefined
inv "верхняя граница": high.oclIsUndefined and highClosed.oclIsUndefined
```

7.10.13 Тип данных PIVL (периодический интервал)

7.10.13.1 Описание

Специализация типа данных QSET

Повторяющийся интервал времени. Тип данных PIVL имеет два свойства: фазу (phase) и период/частоту (period/frequency). Свойство phase задает «прототип интервала», который повторяется с частотой, заданной свойствами period или frequency.

7.10.13.2 Синтаксис ИСО/МЭК 11404

```
type PIVL (T : ANY) = class {
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    originalText : ED.TEXT,
    phase : IVL(TS),
    period : PQ,
    frequency : RTO,
    alignment : CalendarCycle,
    isFlexible : boolean,
    count : INT.POS
}
```

7.10.13.3 Атрибуты

7.10.13.3.1 phase : IVL(TS): прототип повторяющегося интервала, задающий длительность каждого повторения и привязку последовательности экземпляров, задаваемых значением типа PIVL, к определенному моменту времени. Атрибут phase также определяет точку отсчета для всей серии периодически повторяющихся интервалов. Он не может быть пустым или иметь причину пустоты nullFlavor, повторение интервалов не имеет ни начала, ни конца, оно бесконечно как в будущем, так и в прошлом.

Длина интервала (width), задаваемого атрибутом phase, должна быть меньше или равна значению атрибута периода period.

7.10.13.3.2 period : PQ: промежуток времени, являющийся обратной величиной частоты повторения интервалов.

7.10.13.3.3 frequency : RTO: число повторений интервалов (заданное в числителе numerator) в течение определенного периода времени (заданного в знаменателе denominator). Числитель numerator является целым числом, а знаменатель denominator имеет тип PQ.TIME.

Должен задаваться только один из атрибутов period и frequency. Выбранная форма должна наиболее естественно отражать идею для человека, например, каждые 10 мин. (period) или дважды в день (frequency).

7.10.13.3.4 alignment : CalendarCycle: указывает, привязаны ли повторения к календарному циклу и если да, то каким образом (например, чтобы различать «каждые 30 дней» от «5-го числа каждого месяца»). Непривязанное значение типа PIVL повторяется независимо от календаря. Привязанное значение типа PIVL синхронизируется с календарем.

Если этот атрибут имеет значение, то оно должно быть взято из системы кодирования CalendarCycle, определенной рабочей группой HL7. Текущие значения приведены в таблице 27.

Таблица 27 — Перечисление CalendarCycle, ОИД: 2.16.840.1.113883.5.9

Уровень	Код	Описание	Определение
1	CY	year (год)	
1	MY	month of the year (месяц года)	
1	CM	month (continuous) (непрерывный месяц)	
1	CW	week (continuous) (непрерывная неделя)	
1	WM	week of the month (неделя месяца)	
1	WY	week of the year (неделя года)	
1	DM	day of the month (день месяца)	
1	CD	day (continuous) (непрерывный день)	
1	DY	day of the year (день года)	
1	DW	day of the week (begins with Monday) (день недели, начиная с понедельника)	
1	HD	hour of the day (час дня)	
1	CH	hour (continuous) (непрерывный час)	
1	NH	minute of the hour (минута часа)	
1	CN	minute (continuous) (непрерывная минута)	
1	SN	second of the minute (секунда минуты)	
1	CS	second (continuous) (непрерывная секунда)	

7.10.13.4 Синтаксис ИСО/МЭК 11404 атрибута alignment

```
type CalendarCycle = enumeration (CY, MY, CM, CW, WY, DM, CD, DY, DW, HD, CH,
    NH, CN, SN, CS)
```

7.10.13.4.1 isFlexible : Boolean: указывает, оставляется ли точная привязка времени на усмотрение исполнителя расписания, например, чтобы отличить «каждые 8 ч» от «три раза в день».

Примечание — Иногда такая привязка называется «во время, назначенное конкретным учреждением».

7.10.13.4.2 count : INT.POS: общее число повторений периода. Если атрибут count пуст или имеет причину пустоты nullFlavor, то период повторяется бесконечно как до, так и после точки отсчета, заданной атрибутом phase.

7.10.13.5 Инварианты:

- если экземпляр типа данных PIVL не имеет причины пустоты nullFlavor, то может быть указан либо атрибут period, либо атрибут frequency;
- длина width, указанная в атрибуте phase, должна быть меньшей или равной значению атрибута period.

Определение инвариантов на языке OCL:

```
inv "у атрибутов типа данных PIVL нет истории или режима изменений":
    noUpdateOrHistory(phase)and noUpdateOrHistory(period)
inv "no updateMode or History on PIVL attributes":
    noUpdateOrHistory(phase) and noUpdateOrHistory(period)
inv "длина фазы": isNotNull implies
    ((phase.isNotNull implies phase.width < x.period) or
      (frequency.isNotNull implies phase.width <
            (frequency.denominator / frequency.numerator)))
```

7.10.13.6 Примеры

7.10.13.6.1 Дважды в день

```
<example xsi:type='PIVL_TS' isFlexible='true'>
   <period value='12' unit='h'/>
</example>
```

Дважды в день (код BID). Конкретное время оставляется на усмотрение исполнителя расписания.

Это же расписание может быть следующим образом представлено с использованием атрибута frequency:

```
<example xsi:type='PIVL_TS' isFlexible='true'>
    <frequency>
      <numerator xsi:type="INT" value='2'/>
      <denominator xsi:type="PQ" value="1" unit='d'/>
    </frequency>
</example>
```

Этот пример также описывает периодический интервал «дважды в день» (код BID). В простых случаях, наподобие этого, обе эти формы легко преобразуются друг в друга, и человеку обе формы будут равно приемлемы. Хотя преобразование из одной формы в другую всегда возможно, читатель может предпочесть одну или другую в зависимости от конкретных значений.

```
<example xsi:type='PIVL_TS' isFlexible='true'>
    <frequency>
      <numerator xsi:type="INT" value='7'/>
      <denominator xsi:type="PQ" value="1" unit='d'/>
    </frequency>
</example>
```

В этом примере указано, что надо что-то делать семь раз в день. Это же указание с помощью периода воспринимается гораздо сложнее:

```
<example xsi:type='PIVL_TS' isFlexible='true'>
    <period value='3.4285714285714285714285714285714' unit='h'/>
</example>
```

Хотя такой пример может показаться искусственным, подобные ему случаи возникают в клинической практике по всему миру.

7.10.13.6.2 Дважды в день в течение 10 мин.

```
<example xsi:type='PIVL_TS'>
   <phase>
      <width xsi:type="PQ" value='10' unit='min'/>
   </phase>
   <period value='12' unit='h'/>
</example>
```

Дважды в день (каждые 12 ч) в течение 10 мин.

7.10.13.6.3 Каждый сентябрь

```
<example xsi:type='PIVL_TS' alignment='MY'>
    <phase highClosed='true' lowClosed='false'>
      <low value='198709'/>
      <high value='198710'/>
    </phase>
```

```
    <period value='1' unit='a'/> <!-"a" означает год в системе кодирования UCUM
  >
</example>
```

Этот пример несколько сложнее. Он означает сентябрь каждого года (следует учесть, что указание 1987 года в данном примере не влияет на спецификацию периодического интервала, поскольку этот интервал повторяется каждый год и в прошлом, и в будущем).

7.10.13.6.4 Каждую вторую субботу

```
<example xsi:type='PIVL_TS' alignment='DW'>
   <phase highClosed='true' lowClosed='false'>
      <low value='20001202' />
      <high value='20001203' />
   </phase>
   <period value='2' unit='wk'/>
</example>
```

7.10.13.6.5 Повторять через каждые 4—6 ч

```
<example xsi:type='PIVL_TS'>
    <period value='5' unit='h' uncertaintyType='U'>
      <uncertainty value='0.57735' unit='h'/>
    </period>
</example>
```

7.10.14 Тип данных EIVL (периодический интервал времени, связанный с событиями)

7.10.14.1 Описание

Специализация типа данных QSET

Задает периодический интервал времени, повторения которого основаны на событиях повседневной деятельности или других важных событиях, которые связаны с временем, но точно заданного времени не имеют.

Например, расписание «через час после завтрака» указывает, что интервал начинается через час после завершения завтрака. Предполагается, что завтрак происходит до обеда, но при этом никакое конкретное время завтрака не определено.

7.10.14.2 Синтаксис ИСО/МЭК 11404

```
type EIVL = class (
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    originalText : ED.TEXT,
    event : TimingEvent,
    offset : IVL(PQ)
```

7.10.14.3 Атрибуты

7.10.14.3.1 event : TimingEvent: код повседневной (периодической) деятельности, на основе которой определяется периодический интервал, связанный с событием. В домен значений этого атрибута могут включаться такие события, для которых верно все нижеперечисленное:

- событие обычно происходит на регулярной основе;
- событие означает деятельность в течение некоторого времени;
- событие не полностью определяется временем.

Если эти критерии не выполнены, то связь между событием и временем его наступления может передаваться, используя структуры, принадлежащие к типам данных, не описанным в настоящем стандарте.

Если этот атрибут имеет значение, то оно должно быть взято из системы кодирования TimingEvent, определенной рабочей группой HL7. Текущие значения приведены в таблице 28.

Таблица 28 — Перечисление TimingEvent, ОИД: 2.16.840.1.113883.5.139

Уровень	Код	Описание	Определение
1	HS	HS	Часы сна
1	WAKE	WAKE	По пробуждении
1	AC	AC	Перед едой (лат. ante cibus)
2	ACM	ACM	Перед завтраком (лат. ante cibus matutinus)
2	ACD	ACD	Перед обедом (лат. ante cibus diurnus)
2	ACV	ACV	Перед ужином (лат. ante cibus vespertinus)
1	IC	IC	Между приемами пищи (лат. inter cibus)
2	ICM	ICM	Между завтраком и обедом
2	ICD	ICD	Между обедом и ужином
2	ICV	ICV	Между ужином и часами сна
1	PC	PC	После еды (лат. post cibus)
2	PCM	PCM	После завтрака (лат. post cibus matutinus)
2	PCD	PCD	После обеда (лат. post cibus diurnus)
2	PCV	PCV	После ужина (лат. post cibus vespertinus)
1	C	C	Во время еды (лат. cibus)
2	CM	CM	Во время завтрака (лат. cibus matutinus)
2	CD	CD	Во время обеда (лат. cibus diurnus)
2	CV	CV	Во время ужина (лат. cibus vespertinus)

Синтаксис ИСО/МЭК 11404 атрибута event:

```
type TimingEvent = enumeration (HS, WAKE, AC, ACM, ACD, ACV, IC, ICM, ICD,
    ICV, PC, PCM, PCD, PCV, C, CM, CD, CV)
```

7.10.14.3.2 offset : IVL(PQ): интервал прошедшего времени (длительность, а не абсолютный момент времени), задающий смещения начала, длины и конца значения типа EIVL относительно времени фактического совершения такого события.

Например, для расписания «за один час до завтрака в течение 10 минут» атрибут code имеет значение CM, атрибут IVL.low имеет свойство offset, равное – 1 ч, а значение IVL.high имеет свойство offset, равное – 50 мин.

Атрибут offset должен быть пустым, если код события (code) означает «перед едой», «после еды» или «между приемами пищи». Он должен быть непустым, если значение типа EIVL не пусто и код события равен C, CM, CD или CV. Атрибут offset может быть или не быть пустым либо иметь причину пустоты nullFlavor, если код события равен HS или WAKE.

7.10.14.4 Инварианты:

- если значение типа EIVL не имеет причины пустоты nullFlavor, то атрибут event должен быть указан.

Определения инвариантов на языке OCL:

```
inv "требуемые атрибуты": isNotNull implies
    (event.oclIsDefined)
inv "у атрибутов типа данных EIVL нет истории или режима изменений":
     noUpdateOrHistory(offset)
```

7.10.14.5 Примеры

```
<example xsi:type='EIVL_TS' event='CM'>
    <offset>
      <low value='-1' unit='h'/>
      <high value='+50' unit='min'/>
    </offset>
</example>
```

За 1 ч до завтрака в течение 10 мин.

```
<example xsi:type='EIVL_TS' event='CV'>
    <offset>
      <low value='30' unit='min'/>
      <high value='30' unit='min'/>
    </offset>
</example>
```

Через 30 мин. после ужина.

7.10.15 Тонкость GTS.BOUNDEDPIVL

7.10.15.1 Описание

Ограничение типа данных QSI

Тонкость I GTS.BOUNDEDPIVL ограничивает тип данных QSI(TS) таким образом, что допускает пересечение только интервалов типа IVL(TS) и PIVL(TS).

7.10.15.2 Инварианты:

- должно быть два термина;
- один термин должен иметь тип данных IVL(TS);
- другой термин должен иметь тип PIVL(TS);
- атрибут длины width значения типа IVL должен быть пустым (то есть должны присутствовать либо атрибут low, либо атрибут high, либо оба этих атрибута).

Определения инвариантов на языке OCL:

```
inv "GTS.BOUNDEDPIVL 1": terms->size = 2
inv "GTS.BoundedPIVL 2": terms.item >exists(t |
    t.oclIsKindOf(IVL(TS)))
inv "GTS.BoundedPIVL 3": terms.item >exists(t |
    t.oclIsKindOf(PIVL(TS)))
```

7.11 Типы данных, описывающие неопределенность

Эти типы данных предназначены для описания значений, имеющих неопределенность. Следует учесть, что описанные здесь понятия неопределенности, как и в случае типа данных QTY, относятся к неопределенности количества, а не к медицинским понятиям неопределенности, встречающимся в клинической практике, например, «похоже, что имеет место х», или к дифференциальным диагнозам (см. рисунок 12).

7.11.1 Тип данных UVP (неопределенное значение — вероятностное)

7.11.1.1 Описание

Специализация типа данных ANY

Параметр: T : ANY

Общее расширение типа данных, используемое, чтобы указать вероятность, отражающую степень доверия поставщика данных к их значениям.

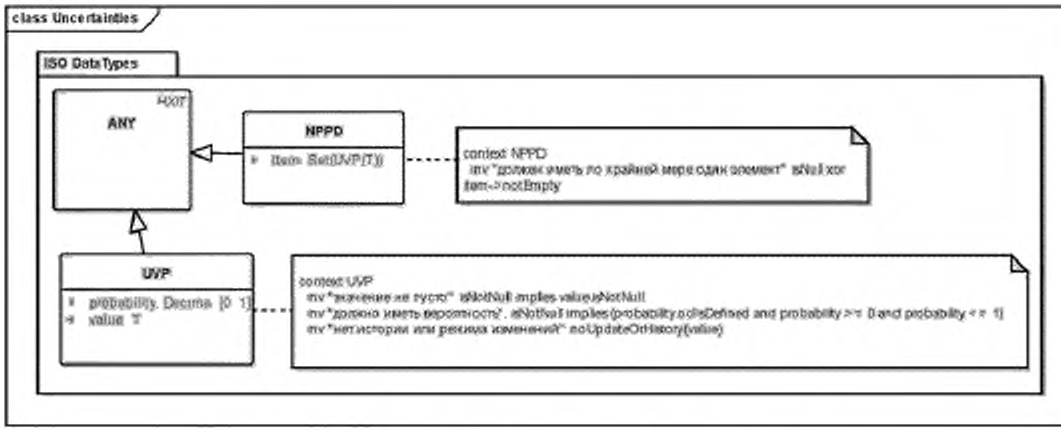


Рисунок 12 — Типы данных, описывающие неопределенность

7.11.1.2 Синтаксис ИСО/МЭК 11404

```
type UVP (T : ANY) = class {
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    probability : Decimal,
    value : T
}
```

7.11.1.3 Атрибуты

7.11.1.3.1 probability : Decimal: вероятность, присвоенная значению, десятичное число от 0 (невозможное) до 1 (определенное) включительно.

Значение вероятности «по умолчанию», которое можно было бы использовать, если вероятность на задана, не существует. Поэтому невозможно провести семантическое различие между значением типа UVP, вероятность которого не указана, и значением простого типа T. Использование типа данных UVP не означает «неопределенность», а использование простого типа данных T — «определенность». Действительно, вероятность значения типа UVP может составлять 0,999 или 1, то есть это значение имеет высокую степень определенности, в то время как значение типа T может быть очень смутной догадкой.

7.11.1.3.2 value : T: значение типа T, к которому относится вероятность.

7.11.1.4 Равенство

Два непустых экземпляра типа данных UVP равны, если попарно равны их атрибуты probability и value.

7.11.1.5 Инварианты:

- атрибут value должен иметь значение;
- атрибут probability должен иметь значение;
- значение атрибута probability должно быть между 0 и 1.

Определение инвариантов на языке OCL:

```
inv "значение не пусто ": isNotNull implies value.isNotNull
  inv "должно иметь вероятность": isNotNull implies
       (probability.oclIsDefined and probability >= 0
       and probability <= 1)
  inv "нет истории или режима изменений": noUpdateOrHistory(probability)
       and noUpdateOrHistory(value)
```

7.11.2 Тип данных NPPD (непараметрическое распределение вероятности)

7.11.2.1 Описание

Специализация типа данных ANY

Параметр: T : ANY

Множество значений типа UVP с вероятностями (известное как гистограмма). Все элементы множества рассматривают как альтернативные и ранжируют по своей вероятности, выражающей степень веры (или частоту) каждого из этих значений.

Тип данных NPPD<T> может быть использован, если только одно значение типа T может быть истинным. Сумма вероятностей должна быть <= 1, но из-за приближенных вычислений и неточностей, связанных с округлением, эта сумма может превысить 1.

7.11.2.2 Синтаксис ИСО/МЭК 11404

```
type NPPD (T : ANY) = class {
    validTimeLow : characterstring,
    validTimeHigh : characterstring,
    controlInformationRoot : characterstring,
    controlInformationExtension : characterstring,
    nullFlavor : NullFlavor,
    updateMode : UpdateMode,
    flavorId : Set(characterstring),
    item : Set(UVP(T))
}
```

7.11.2.3 Атрибуты

7.11.2.3.1 item : Set(UVP(T)): список значений с вероятностями, представляемый гистограммой.

7.11.2.4 Равенство

Два непустых экземпляра типа данных NPPD равны, если они содержат одни и те же элементы.

Примечание — Определение содержания элемента основано на том же семантическом равенстве, которое определено в настоящем стандарте, поэтому, к примеру, значение типа NPPD(CD) может быть равно значению типа NPPD(CS).

7.11.2.5 Инварианты:

- должно присутствовать по меньшей мере одно значение.

Определение инвариантов на языке OCL:

```
inv "должен иметь по меньшей мере один элемент": isNull xor item->notEmpty
```

7.11.2.6 Примеры

```
<example xsi:type='NPPD_ST'>
    <item probability="0.1">
      <value value="Yankees"/>
    </item>
    <item probability="0.04">
      <value value="Red Sox"/>
    </item>
    <item probability="0.05">
      <value value="White Sox"/>
    </item>
    <item probability="0.08">
      <value value="Indians"/>
    </item>
    <item probability="0.05">
      <value value="Tigers"/>
    </item>
    <item probability="0.07">
      <value value="Mariners"/>
```

```
    </item>
    <item probability="0.02">
      <value value="Royals"/>
    </item>
    <item probability="0.06">
      <value value="Orioles"/>
    </item>
</example>
```

7.12 Структурированный текст

7.12.1 Введение

В этом подразделе описаны типы данных SD.TEXT и SD.TITLE. Оба этих типа данных представляют структуры, подобные документам. В дополнение к тексту они могут содержать мультимедийные данные, списки, таблицы, форматирование, а также информацию о представлении содержания и ссылки.

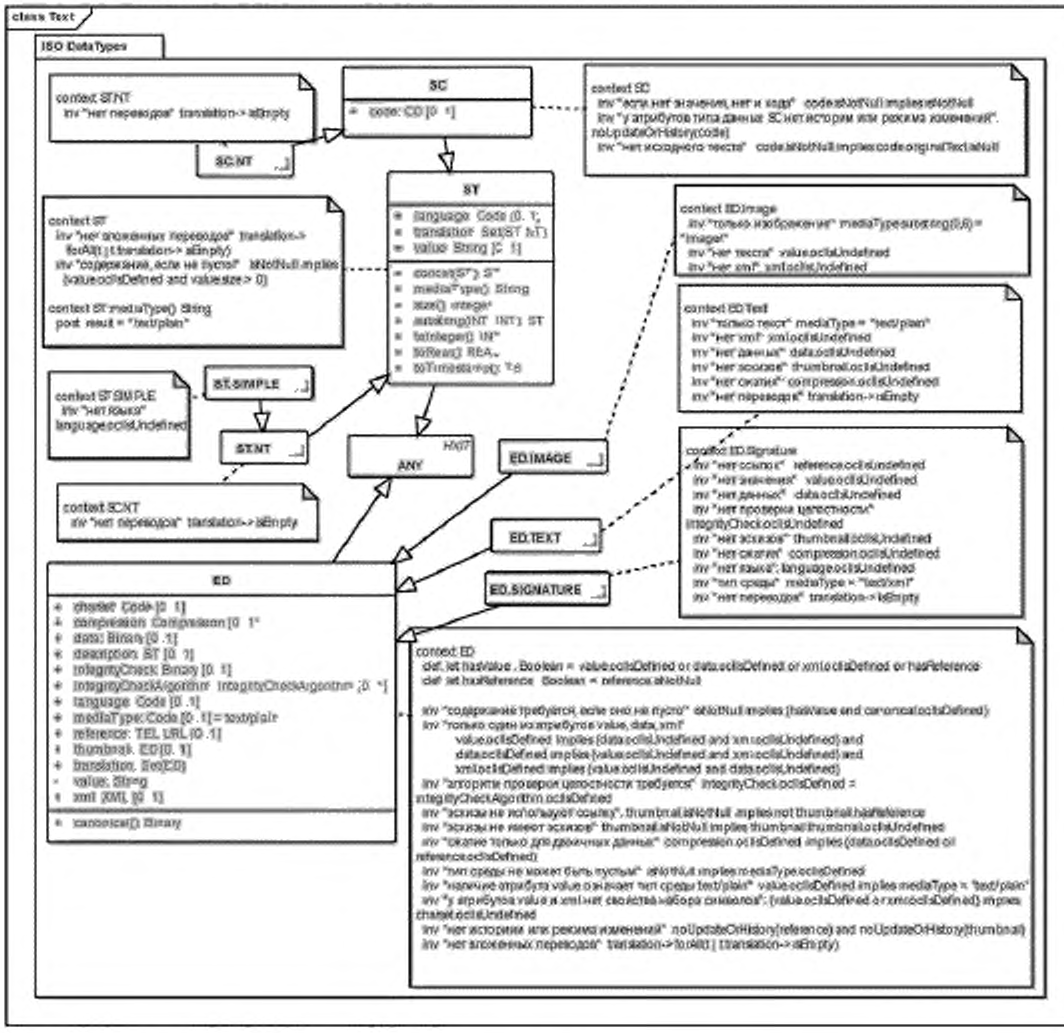


Рисунок 13 — Модель структурированного текста

Эти структуры не рассчитаны на полное представление документов. Они представляют собой строительные блоки, предназначенные для встраивания в больший контекст наряду с другими классами, использующими эти форматы документа и другие типы данных для конструирования полного документа, полезного для применения в клинической практике и в других задачах здравоохранения. Однако эти структуры могут также найти применение в других форматах записей и сообщений. Будь то документы, записи или сообщения, они создаются внутри единого контекста. Такой контекст документов может содержать несколько структур типов, описанных в настоящем подразделе, и эти структуры могут ссылаться на любое содержание внутри этого единого контекста документов.

Любой элемент обработки информации, объявляющий непосредственное или косвенное соответствие настоящему стандарту и поддерживающий использование структур формата документов, описанных в настоящем стандарте, должен документировать вид используемого контекста и точно определять, каким образом разрешаются ссылки внутри контекста документов.

В этом подразделе описана специальная поддержка связи изменений содержания с предыдущими версиями содержания. Если она обеспечивается, то объявление соответствия должно точно указывать, каким образом доступна релевантная информация о версии и как она интегрирована с контролем изменения содержания.

Такие документы предназначены для «отображения». Документ отображается, когда он готовится для восприятия человеком на экране монитора, в виде напечатанного отчета или каким-либо иным методом. Отображение документа должно следовать правилам, описанным в настоящем подразделе.

Такие структуры, подобные документам, имеют очевидное сходство со спецификацией языка XHTML, а их функции в известной мере перекрывают функции языка XHTML. Однако эти структуры имеют некоторые очень существенные концептуальные отличия от языка XHTML, особенно в части интеграции этих структур в их контекст. Для языка XHTML контекстом служит Всемирная паутина? и он обладает свойствами, которые тесно связывают его с протоколом, в то время как описываемые структуры предназначены для использования в XML-документах, которые могут содержать несколько таких структур и двунаправленные внутренние и внешние ссылки. Кроме того, эти структуры имеют некоторые дополнительные функциональные характеристики, в основном те, которые связаны с использованием типов данных StrucDoc.Content и RenderMultimedia. В конкретном контексте использования эти структуры без труда могут быть преобразованы в код на языке XHTML.

Функционально эти структуры подобны типу данных ED и могут быть преобразованы в атрибут ED.data. Подобно типу данных ED, потомки типа данных SD имеют содержание на языке xml, набор символов, язык и могут иметь атрибут причины пустоты nullFlavor. Другие свойства типа данных ED, например, ссылка, проверка целостности, эскиз и переводы, должны иметь фиксированное пустое значение.

7.12.2 Пример

```
<example xsi:type="SD.TEXT">
    <paragraph>
      <caption styleCode="Bold xHead1">Introduction</caption>
      Thank you for referring this patient for investigation
      into <content ID="c1">burnt ears</content>.
    </paragraph>
    <list>
      <caption styleCode="Bold xHead1">Initial Observations</caption>
      <item>The patient presented in a very confused state.</item>
      <item>
            There was extensive damage to the outer ears:
            <renderMultiMedia referencedObject="il">
                  <caption>Photo of left ear</caption>
            </renderMultiMedia>
      </item>
    </list>
    <table summary="Investigations performed"   border="all" rules="all">
      <caption styleCode="Bold xHead1">Investigations</caption>
      <thead>
            <tr>
                  <th>Investigation</th><th>Finding</th>
```

```
            </tr>
      </thead>
      <tbody>
            <tr>
                  <th><content ID="c2">Skin Condition</content></th>
                  <th><content ID="c3">1<sup>st</sup> degree burns</content></
th>
            </tr>
            <tr>
                  <th><content ID="c4">Hearing Test</content></th>
                  <th><content ID="c5">The patients hearing is okay</content></
th>
            </tr>
      </tbody>
    </table>
    <paragraph>
      <caption styleCode="Bold xHead1">Recommendations</caption>
      The patient should apply a cream to the outer ears until
      they are healed. <content revised="insert">The patient
      should wear a woollen balaclava in the future while ironing
      his shirts to prevent a re occurence of the accident<footnote>This
      has been proven to offer the best protection against a repeat
      injury. See <linkHtml href="                                    ">
            the wikipedia article about burnt ears</linkHtml> for further
      information.</footnote>.</content>
    </paragraph>
</example>
```

gost.e-ecolog.ru

Примечания

1 Этот документ имеет четыре раздела: введение (introduction), первичный осмотр (initial observations), результаты обследования (investigations) и рекомендации (recommendations). Хотя этот пример и не является серьезным, подобный контекст использования структурированного текста может задавать специфичные правила семантики его конкретных разделов, тем самым ставя для них более узкие рамки.

2 Код xHead1 стиля styleCode служит примером допустимого местного расширения кодов стилей.

3 Элемент content используется, чтобы задать атрибут ID. Значение этого атрибута может использоваться в элементе ссылки originalText.reference, входящем в значение типа CD или PQ, включенное в другое содержание и ссылающееся на фрагмент структурированного текста.

4 Последний экземпляр элемента content обозначает исправление документа.

5 В контексте данного документа должен присутствовать некоторый элемент, у которого атрибут ID имеет значение «i1». Им должна быть разновидность класса, представляющего некоторое мультимедийное содержание.

6 Хотя фактический процесс представления не входит в область применения настоящего стандарта, на рисунке 14 показана одна из возможных форм представления данного документа.

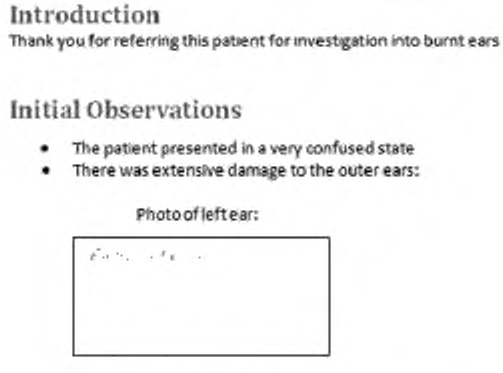
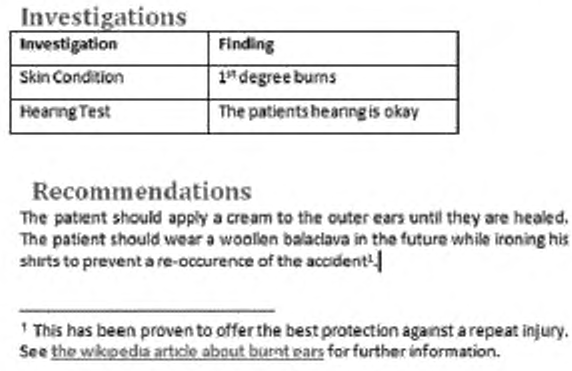


Рисунок 14, лист 1 — Форма представления



Investigations

Investigation	Finding
Skin Condition	1st degree burns
Hearing Test	The patients hearing is okay

Recommendations

The patient should apply a cream to the outer ears until they are healed.
The patient should wear a woollen balaclava in the future while ironing his shirts to prevent a re-occurence of the accident[1].

[1] This has been proven to offer the best protection against a repeat injury. See the wikipedia article about burnt ears for further information.

Рисунок 14, лист 2

7.12.3 Тип данных StrucDoc.Base

7.12.3.1 Описание

Абстрактный тип данных

Задает базовую идентификацию и атрибуты стиля представления, используемые многими элементами структурированного документа.

7.12.3.2 Атрибуты

7.12.3.2.1 ID : String: уникальный идентификатор данного элемента в документе.

7.12.3.2.2 language : Code: язык элемента. Дополнительная информация приведена в 7.4.2.3.7. В документе указание языка применяется ко всем вложенным элементам, если только в них специально не указан другой язык.

7.12.3.2.3 styleCode : Set(Code): стили представления, применяемые к данному документу.

Если этому атрибуту присваивается значение, то оно должно быть либо одним из кодов, приведенных в таблице 29, либо допустимым местным расширением.

Таблица 29 — Перечисление StyleCode

Уровень	Код	Определение
1	Стиль шрифта (задает характеристики отображения шрифта)	
2	Bold	Отображать полужирным шрифтом
2	Underline	Отображать подчеркнутым шрифтом
2	Italics	Отображать курсивом
2	Emphasis	Отображать некоторым типом выделения
1	Стиль обрамления таблицы (задает характеристики отображения ячеек таблицы)	
2	Lrule	Отображать левую границу ячейки
2	Rrule	Отображать правую границу ячейки
2	Toprule	Отображать верхнюю границу ячейки
2	Botrule	Отображать нижнюю границу ячейки
1	Стиль упорядоченного списка (задает характеристики отображения упорядоченных списков)	
2	Arabic	Нумерация арабскими цифрами: 1, 2, 3
2	LittleRoman	Нумерация римскими цифрами в нижнем регистре: i, ii, iii
2	BigRoman	Нумерация римскими цифрами в верхнем регистре: I, II, III
2	LittleAlpha	Нумерация буквами в нижнем регистре: a, b, c

Окончание таблицы 29

Уровень	Код	Определение
2	BigAlpha	Нумерация буквами в верхнем регистре: A, B, C
1	Стиль не упорядоченного списка (задает характеристики отображения не упорядоченных списков)	
2	Disc	Маркерами списка служат простые сплошные кружки
2	Circle	Маркерами списка служат незаполненные кружки
2	Square	Маркерами списка служат сплошные квадраты

Местные расширения перечисления StyleCode должны удовлетворять следующему шаблону: [x][A-Za-z][A-Za-z0-9]* (первый символ «x», второй — буквы от A до Z в верхнем или нижнем регистре, остальные символы — любое сочетание буквы от A до Z в верхнем или нижнем регистре и цифр).

7.12.4 Тип данных StrucDoc.Br

7.12.4.1 Описание

Определение: переход на начало следующей строки, подобный используемому в языке HTML.

7.12.5 Тип данных StrucDoc.Sup

7.12.5.1 Описание

Указывает, что текст, содержащийся в атрибуте text, должен отображаться как верхний индекс, например, x^2.

7.12.5.2 Атрибуты

7.12.5.2.1 text : String «XMLText»: текст, представляемый в виде верхнего индекса.

Значение стереотипа XMLText описано в А.2 приложения А.

7.12.5.3 Инварианты:

- текст не должен быть пустым.

Представление инвариантов на языке OCL:

```
inv "текст не должен быть пустым": text.length > 0
```

7.12.6 Тип данных StrucDoc.Sub

7.12.6.1 Описание

Указывает, что текст, содержащийся в атрибуте text, должен отображаться как нижний индекс, например, H_2O.

7.12.6.2 Атрибуты

7.12.6.2.1 text : String «XMLText»: текст, представляемый в виде нижнего индекса.

Значение стереотипа XMLText описано в А.2 приложения А.

7.12.6.3 Инварианты:

- текст не должен быть пустым.

Представление инвариантов на языке OCL:

```
inv "текст не должен быть пустым": text.length > 0
```

7.12.7 Тип данных StrucDoc.LinkHtml

7.12.7.1 Описание

Специализация типа данных StrucDoc.Base

Гипертекстовая ссылка на другой документ. Такие ссылки обычно отображаются специальным образом, позволяющим пользователю активировать их при просмотре документа.

Функциональность ссылки предоставлена общим механизмом ссылок, подобным, но не идентичным тегу якоря <a> на языке HTML. Она может быть использована для ссылки на идентификаторы, являющиеся как внутренними, так и внешними по отношению к документу или контексту документа.

Для ссылки на мультимедийные данные, не являющиеся частью документа, может быть использована обычная гипертекстовая ссылка. Не требуется, чтобы получатель отображал внутреннюю или внешнюю ссылку либо цель внешней ссылки.

7.12.7.2 Атрибуты

7.12.7.2.1 href : String: адрес URL, идентифицирующий целевой документ или объект ссылки. Целью является идентификатор XML, внутренний или внешний по отношению к документу. В контексте использования должна быть точно определена область разрешения ссылки. Следуя соглашениям, принятым в языке HTML, внутренняя ссылка (обычно ссылка в пределах контекста документа) должна иметь префикс в виде символа «#».

7.12.7.2.2 rel : Set(StrucDoc.LinkType): этот атрибут описывает отношения между текущим документом и якорем, заданным атрибутом href. Значение атрибута rel представляет собой список типов ссылок, разделенных пробелами.

Если этому атрибуту присвоено значение, то указанные в нем типы ссылок должны браться из перечисления StyleCode, приведенного в таблице 30.

Этот список взят из спецификации языка HTML.

Таблица 30 — Перечисление StyleCode

Уровень	Код	Определение
1	Alternate	Обозначает заменяющие версии документа, в котором указана ссылка
1	Stylesheet	Ссылка на внешнюю таблицу стилей. Используется вместе с типом ссылки «Alternate» для ссылки на выбираемые пользователем альтернативные таблицы стилей
1	Start	Ссылка на первый документ в коллекции документов
1	Next	Ссылка на следующий документ в линейном списке документов. Пользовательские агенты могут осуществлять упреждающую загрузку «следующего» документа, чтобы сократить время ее ожидания
1	Prev	Ссылка на предшествующий документ в упорядоченном списке документов. Некоторые пользовательские агенты поддерживают также синоним «Previous»
1	Contents	Ссылка на документ, который служит оглавлением текущего документа. Некоторые пользовательские агенты поддерживают также синоним ToC (от «Table of Contents» — оглавление)
1	Index	Ссылка на документ, служащий предметным указателем текущего документа
1	Glossary	Ссылка на документ, служащий глоссарием текущего документа
1	Copyright	Ссылка на объявление авторских прав на текущий документ
1	Chapter	Ссылка на документ, представляющий собой главу в коллекции документов
1	Section	Ссылка на документ, представляющий собой раздел в коллекции документов
1	Subsection	Ссылка на документ, представляющий собой подраздел в коллекции документов
1	Appendix	Ссылка на документ, представляющий собой приложение в коллекции документов
1	Help	Ссылка на поясняющий документ (дополнительная информация, ссылки на другие источники информации и т. д.)
1	Bookmark	Ссылка на закладку. Закладка обозначает ключевую точку входа в расширенный документ. Для именования закладки может быть использован, к примеру, атрибут title. Учитите, что в каждом документе может быть определено несколько закладок

7.12.7.2.3 rev : Set(StrucDoc.LinkType): этот атрибут используется для описания обратной ссылки на якорь, заданный в текущем документе атрибутом href. Значение атрибута rel представляет собой список типов ссылок, разделенных пробелами. Допустимые значения приведены в таблице 30.

7.12.7.2.4 title : String: в этом атрибуте указано краткое описание содержащего его элемента. Атрибут title имеет дополнительную роль при использовании вместе с элементом LINK, обозначающим внешнюю таблицу стилей. Дополнительная информация приведена в стандарте языка HTML.

Примечание — Пользовательские агенты могут отображать значения атрибута title разными способами. Например, визуальные обозреватели изображают его значение как «подсказку» (краткое сообщение, которое появляется, когда указатель мыши или аналогичного устройства наводится на объект). Речевые пользовательские агенты могут в аналогичном контексте зачитывать эту информацию. Например, включение этого атрибута в элемент ссылки позволяет пользовательским агентам (визуальным и не визуальным) сообщать пользователям краткую информацию о ресурсе, являющемся целью ссылки.

7.12.7.3 Ассоциации

7.12.7.3.1 parts : CMFootnotes [0..* упорядоченные] «Anonymous»: содержание (текст и сноски), представляющее текст, с которым ассоциирована активируемая ссылка.

Значение стереотипа Anonymous описано в А.2 приложения А.

7.12.7.4 Инварианты:

- со ссылкой должен быть ассоциирован некоторый текст.

Описание инвариантов на языке OCL:

```
inv "должен иметься по крайней мере один элемент": parts->notEmpty
```

7.12.7.5 Пример

```
<text>История заболевания коронарных артерий, описанная
    <linkHtml href="#SECT001">выше</linkHtml>.
</text>
```

Явная ссылка. В данном примере приведена ссылка на раздел CDA-документа с идентификатором «SECT001»:

```
<section ID="SECT001">
    <code code="10153-2" codeSystem="2.16.840.1.113883.6.1"
      codeSystemName="LOINC"/>
    <title>Past Medical History</title>
</section>
```

7.12.8 Тип данных StrucDoc.RenderMultiMedia

7.12.8.1 Описание

Специализация типа данных StrucDoc.Base

Указывает ссылку на мультимедийные данные, вложенные в документ, и служит для обозначения места, где эти данные должны быть отображены. Мультимедийные данные должны содержаться в контексте документа.

У этого типа данных есть необязательный атрибут заголовка caption и требуемый атрибут referencedObject (типа XML IDREFS), значения которого должны быть равны значениям идентификаторов XML ID разделов ObservationMedia или RegionOfInterest, содержащихся в контексте CDA-документа.

7.12.8.2 Атрибуты

7.12.8.2.1 caption : StrucDoc.Caption: необязательный заголовок мультимедийных данных.

7.12.8.2.2 referencedObject: Set(String) «XMLIDREF»: ссылки на другие идентифицированные объекты, содержащиеся в контексте документа.

Значение стереотипа XMLIDREF описано в А.2 приложения А.

7.12.8.3 Инварианты:

- должна быть указана по меньшей мере одна ссылка.

Представление инвариантов на языке OCL:

```
inv "должен иметь по меньшей мере одну ссылку": referencedObject->size() > 0
```

7.12.9 Тип данных StrucDoc.FootnoteRef

7.12.9.1 Описание

Специализация типа данных StrucDoc.Base

Ссылка на существующую сноску в контексте документа. На одну и ту же сноску можно ссылаться несколько раз. Значение footnoteRef.IDREF должно совпадать со значением footnote.ID в том же самом документе.

7.12.9.2 Атрибуты

7.12.9.2.1 IDREF : String «XMLIDREF»: идентификатор сноски, на которую дается ссылка.

Значение стереотипа XMLIDREF описано в А.2 приложения А.

7.12.9.3 Инварианты:

- ссылка должна иметь значение.

Определение инвариантов на языке OCL:

```
inv "должна иметь значение": IDREF.oclIsDefined
```

7.12.10 Тип данных StrucDoc.Footnote

7.12.10.1 Описание

Специализация типа данных StrucDoc.Base

Обозначает сноску. Содержание значения Footnote является содержанием сноски. При отображении документа ссылка на сноску изображена в одной строке с текстом, примыкающим к сноске.

Примечание — Получатели должны интерпретировать эти элементы при отображении визуально отличающегося текста сноски. Точное представление оставляется на усмотрение получателя и может состоять в знаке сноски, представляющем собой гиперссылку на текст сноски, в простом выделении текста сноски (например, «Это текст [это сноска], к которому сделана сноска»), и т. д.

7.12.10.2 Ассоциации

1.1.3.3.1 parts : StrucDoc.CMGeneral[0..* [0..* упорядочен] «Anonymous»: содержание сноски.

Значение стереотипа Anonymous описано в А.2 приложения А.

7.12.10.3 Инварианты:

- сноска должна иметь некоторое содержание;
- сноски не могут содержать вложенные сноски.

Определение инвариантов на языке OCL:

```
inv "требуется некоторое содержание": parts->notEmpty
inv "нет вложенных сносок": parts->forAll(t |
    t.footnote.oclIsUndefined and t.footnoteRef.oclIsUndefined)
```

7.12.11 Тип данных StrucDoc.TitleFootnote

7.12.11.1 Описание

Специализация типа данных StrucDoc.Base

Та же функциональность, что у обычной сноски, но модель содержания атрибута parts ограничена типом содержания, которое может появиться в заголовке.

7.12.11.2 Ассоциации

7.12.11.2.1 parts : StrucDoc.CMTitle[0..* упорядочен] «Anonymous»: содержание сноски.

Значение стереотипа Anonymous описано в подразделе А.2.

7.12.11.3 Инварианты:

- сноска должна иметь некоторое содержание.

Определение инвариантов на языке OCL:

```
inv "требуется некоторое содержание": parts->notEmpty
```

7.12.12 Тип данных StrucDoc.Content

7.12.12.1 Описание

Специализация типа данных StrucDoc.Base.

Используется в качестве обертки фрагмента текста, чтобы на него можно было явно сослаться или задать характеристики его отображения. Обертки могут быть вложенными, что позволяет разбить строку обычного текста на сколь угодно малые фрагменты.

Обертка имеет необязательный идентификатор, который может служить целью ссылки. Этот идентификатор, передаваемый в атрибуте XML ID, должен быть уникальным во всем контексте документа. Используя этот идентификатор, атрибут исходного текста (originalText), имеющийся у типов данных, определенных в настоящем стандарте, может содержать явную ссылку на такую обертку и тем самым указывать на исходный текст, ассоциированный со значением такого типа данных.

7.12.12.2 Атрибуты

7.12.12.2.1 revised : Revised: может использоваться для указания последних изменений повествовательного содержания CDA-документа. Этот атрибут ограничен одним поколением, так что может отражать только изменение, сделанное в предыдущей версии документа. Получатели должны интерпретировать атрибут revised при отображении удаленного фрагмента текста, визуально выделяя его или скрывая.

Если применим, этот атрибут должен быть использован в сочетании с соответствующим отслеживанием версий документов, определенным в объявлении соответствия, данном для контекста документов.

Если этому атрибуту присвоено значение, то оно должно браться из перечисления StrucDoc. Revised, приведенного в таблице 31.

Таблица 31 — Перечисление StrucDoc.Revised

Уровень	Код	Описание	Определение
1	insert	Вставить	В этой редакции документа данное содержание было вставлено
1	delete	Удалить	В этой редакции документа данное содержание было удалено

7.12.12.3 Ассоциации

7.12.12.3.1 parts : StrucDoc.CMContent[0..* упорядочен] «Anonymous»: содержание значения типа Content.

Значение стереотипа Anonymous описано в А.2 приложения А.

7.12.12.4 Инварианты:

- требуется некоторое содержание.

Определение инвариантов на языке OCL:

```
inv "требуется некоторое содержание": parts->notEmpt
```

7.12.13 Тип данных StrucDoc.Caption

7.12.13.1 Описание

Специализация типа данных StrucDoc.Base

Название абзаца, списка, элемента списка, таблицы или ячейки таблицы. Оно может также быть использовано в значении типа RenderMultiMedia для обозначения названия разделов ObservationMedia или RegionOfInterest, на которые дается ссылка. Значение типа Caption содержит неформатированный текст и может содержать ссылки и сноски.

Если название определено, то оно должно отображаться и должно быть представлено раньше того элемента, с которым оно ассоциировано.

7.12.13.2 Ассоциации

7.12.13.2.1 parts : StrucDoc.CMInline[0..* упорядочен] «Anonymous»: содержание значения типа Content.

Значение стереотипа Anonymous описано в А.2 приложения А.

7.12.13.3 Инварианты:

- название должно иметь некоторое содержание.

Определение инвариантов на языке OCL:

```
inv "требуется некоторое содержание": parts->notEmpty
```

7.12.14 Тип данных StrucDoc.Captioned

7.12.14.1 Описание

Абстрактный: специализация типа данных StrucDoc.Base

Абстрактный предшественник всех типов, имеющих атрибут названия caption.

Если атрибут caption определен, то он должен отображаться и должен быть представлен раньше того элемента, с которым он ассоциирован.

7.12.14.2 Атрибуты

7.12.14.2.1 caption : StrucDoc.Caption[0..* упорядочен] «Anonymous»: содержание значения типа Content.

Значение стереотипа Anonymous описано в А.2 приложения А.

7.12.15 Тип данных StrucDoc.Paragraph

7.12.15.1 Описание

Специализация типа данных StrucDoc.Captioned

Аналогичен абзацу на языке HTML, позволяющему разбивать повествовательные блоки на логически связанные структуры.

7.12.15.2 Ассоциации

7.12.15.2.1 parts : StrucDoc.CMInline[0..* ordered] «Anonymous»: содержание значения типа Content.

Значение стереотипа Anonymous описано в А.2 приложения А.

7.12.15.3 Инварианты:

- атрибут caption должен иметь некоторое содержание.

Определение инвариантов на языке OCL:

```
inv "требуется некоторое содержание": parts->notEmpty
```

7.12.16 Тип данных StrucDoc.CMFootnotes

7.12.16.1 Описание

Стереотип: «Choice»

Модель содержания, позволяющая текст и сноски. Стереотип Choice означает, что ровно один из атрибутов должен иметь значение, все остальные должны быть пустыми.

7.12.16.2 Атрибуты

7.12.16.2.1 Text : String «XMLText»: неформатированный текст.

Значение стереотипа XMLText описано в А.2 приложения А.

7.12.16.2.2 footnote : StrucDoc.Footnote: текст сноски.

7.12.16.2.3 footnoteRef StrucDoc.FootnoteRef: ссылка на текст сноски.

7.12.17 Тип данных StrucDoc.CMInline

7.12.17.1 Описание

Специализация типа данных StrucDoc.CMFootnotes

Стереотип: «Choice»

Модель содержания, позволяющая текст, сноски, ссылки, верхние и нижние индексы. Стереотип Choice означает, что ровно один из атрибутов (включая наследуемые) должен иметь значение, все остальные должны быть пустыми.

7.12.17.2 Атрибуты

7.12.17.2.1 linkHtml : StrucDoc.LinkHtml: HTML-подобная ссылка.

7.12.17.2.2 sub: StrucDoc.Sub : текст нижнего индекса.

7.12.17.2.3 sup: StrucDoc.Sup : текст верхнего индекса.

7.12.18 Тип данных StrucDoc.CMContent

7.12.18.1 Описание

Специализация типа данных StrucDoc.CMInline

Модель содержания, позволяющая текст, сноски, ссылки, верхние и нижние индексы, переходы на начало следующей строки, мультимедийные данные и вложенные элементы content. Стереотип Choice означает, что ровно один из атрибутов (включая наследуемые) должен иметь значение, все остальные должны быть пустыми.

7.12.18.2 Атрибуты

7.12.18.2.1 content : StrucDoc.Content: вложенная обертка фрагмента текста.

7.12.18.2.2 br: StrucDoc.Br : переход на начало следующей строки.

7.12.18.2.3 renderMultiMedia : StrucDoc.RenderMultiMedia: мультимедийные данные.

7.12.19 Тип данных StrucDoc.CMGeneral

7.12.19.1 Описание

Специализация типа данных StrucDoc.CMContent.

Модель содержания, позволяющая текст, сноски, ссылки, верхние и нижние индексы, переходы на начало следующей строки, мультимедийные данные, вложенные элементы content, абзацы, списки и таблицы. Стереотип Choice означает, что только один из атрибутов (включая наследуемые) должен иметь значение, все остальные должны быть пустыми.

7.12.19.2 Атрибуты

7.12.19.2.1 paragraph : StrucDoc.Paragaph: абзац и другое содержание типа CMContent.

7.12.19.2.2 list : StrucDoc.List: содержание, основанное на списке.

7.12.19.2.3 table : StrucDoc.Table: таблица.

7.12.20 Тип данных StrucDoc.CMTitle

7.12.20.1 Описание

Стереотип: «Choice»

Модель содержания, позволяющая текст, сноски, ссылки, верхние и нижние индексы, переходы на начало следующей строки, и вложенные элементы content. Мультимедийные данные не разрешены. Стереотип Choice означает, что ровно один из атрибутов (включая наследуемые) должен иметь значение, все остальные должны быть пустыми.

7.12.20.2 Атрибуты

7.12.20.2.1 text : String «XMLText»: неформатированный текст.

Значение стереотипа XMLText описано в А.2 приложения А.

7.12.20.2.2 footnote : StrucDoc.Footnote: текст сноски.

7.12.20.2.3 footnoteRef : StrucDoc.FootnoteRef: ссылка на текст сноски.

7.12.20.2.4 br : StrucDoc.Br: переход на начало следующей строки.

7.12.20.2.5 linkHtml : StrucDoc.LinkHtml: HTML-подобная ссылка.

7.12.20.2.6 sub : StrucDoc.Sub: текст нижнего индекса.

7.12.20.2.7 sup : StrucDoc.Sup: текст верхнего индекса.

7.12.20.2.8 content : StrucDoc.Content: вложенная обертка фрагмента текста.

7.12.21 Тип данных StrucDoc.List

7.12.21.1 Описание

Специализация типа данных StrucDoc.Captioned

Подобен списку на языке HTML. Имеет необязательное название и один или несколько элементов. Список должен быть упорядоченным или неупорядоченным, это всегда должно быть известно.

7.12.21.2 Атрибуты

7.12.21.2.1 listType : StrucDoc.ListType: тип списка — упорядоченный или неупорядоченный.

Если этому атрибуту присвоено значение, то оно должно браться из перечисления StrucDoc. ListType, приведенного в таблице 32.

Таблица 32 — Перечисление StrucDoc.Revised

Уровень	Код	Описание	Определение
1	ordered	Упорядоченный	Упорядоченный список
1	unordered	Неупорядоченный	Не упорядоченный список

По умолчанию список считается не упорядоченным.

Примечание — Элементы неупорядоченного списка обычно отображаются маркированными, а элементы упорядоченного списка — нумерованными, но это не является обязательным.

7.12.21.3 Ассоциации

7.12.21.3.1 item : StrucDoc.Item[0..* упорядочен]: фактические элементы списка.

7.12.21.4 Инварианты:

- по меньшей мере один элемент должен иметь значение.

Определение инвариантов на языке OCL:

```
inv "должен иметь хотя бы один элемент": item->notEmpty
```

7.12.22 Тип данных StrucDoc.Item

7.12.22.1 Описание

Специализация типа данных StrucDoc.Captioned

Элемент списка.

7.12.22.2 Ассоциации

7.12.22.2.1 parts : StrucDoc.CMGeneral[0..* упорядочен] «Anonymous»: содержание сноски.

Значение стереотипа Anonymous описано в А.2 приложения А.

7.12.23 Тип данных StrucDoc.TableItem

7.12.23.1 Описание

Абстрактный; специализация типа данных StrucDoc.Base.

Абстрактный контейнер элемента таблицы, в котором могут быть указаны такие детали отображения таблицы, как выравнивание.

Любой атрибут, примененный к элементу таблицы, применяется также и ко всем другим вложенным в него элементам, если специально не будет переопределен в них.

7.12.23.2 Атрибуты

7.12.23.2.1 align : StrucDoc.Align: выравнивание текста, применяемое к ячейке таблицы.

Если этому атрибуту присвоено значение, то оно должно браться из перечисления StrucDoc.CellAlign, приведенного в таблице 33.

Таблица 33 — Перечисление StrucDoc.CellAlign

Уровень	Код	Описание	Определение
1	left	Влево	Содержание выравнивается влево. Этот длинный абзац служит примером выравнивания содержания ячейки влево
1	center	По центру	Содержание выравнивается по центру. Этот длинный абзац служит примером выравнивания содержания ячейки по центру
1	right	Вправо	Содержание выравнивается вправо. Этот длинный абзац служит примером выравнивания содержания ячейки вправо
1	justify	По ширине	Содержание выравнивается по ширине. Этот длинный абзац служит примером выравнивания содержания ячейки по ширине
1	char	Символ	align=char выравнивание содержания ячейки по символу, указанному в атрибуте char

По умолчанию содержание ячейки выравнивается влево.

7.12.23.2.2 char : String: символ, по которому выравнивается содержание ячейки (если задано выравнивание по символу). По умолчанию значение атрибута char полагается равным разделителю десятичной дроби, принятому в текущем языке — в английском языке им служит точка.

7.12.23.2.3 charoff : StrucDoc.Length: если указан, этот атрибут задает смещение первого вхождения символа выравнивания на каждой строке. Если строка не содержит символ выравнивания, то она должна быть горизонтально смещена до конца позиции выравнивания. Поддержка этого атрибута элементами обработки информации не требуется.

7.12.23.2.4 valign : StrucDoc.VAlign: вертикальное выравнивание содержания ячейки.

Если этому атрибуту присвоено значение, то оно должно браться из перечисления StrucDoc.VAlign, приведенного в таблице 34.

Таблица 34 — Перечисление StrucDoc.VAlign

Уровень	Код	Описание	Определение
1	top	По верху	Содержание выравнивается по верху ячейки, как показано в столбце «Описание»
1	middle	По центру	Содержание выравнивается по центру ячейки, как показано в столбце «Описание»
1	bottom	По низу	Содержание выравнивается по низу ячейки, как показано в столбце «Описание»
1	baseline	По базовой линии	Во всех ячейках одной и той же строки таблицы, имеющих атрибут valign с этим значением, текст должен выравниваться таким образом, чтобы первая строка текста находилась на базовой линии, общей для всех этих ячеек. Это требование не распространяется на следующие строки текста, находящиеся в этих ячейках

По умолчанию содержание ячеек выравнивается по верху.

7.12.24 Тип данных StrucDoc.TCell

7.12.24.1 Описание

Специализация типа данных StrucDoc.TableItem

Ячейка таблицы — может быть или обычной ячейкой, или ячейкой заголовка таблицы.

7.12.24.2 Атрибуты

7.12.24.2.1 abbr : String: этот атрибут должен использоваться для представления сокращенного содержания ячейки и может отображаться пользовательскими агентами вместо полного содержания ячейки. Сокращенное содержание должно быть достаточно коротким, поскольку пользовательские агенты могут отображать его повторяющимся.

7.12.24.2.2 axis : String: этот атрибут может использоваться для отнесения ячейки к концептуальной категории, которые могут рассматриваться как оси n-мерного пространства. Пользовательские агенты могут давать пользователям доступ к отдельным категориям (например, пользователь может запросить у агента все ячейки, принадлежащие к определенным категориям, агент может представить таблицу в форме оглавления и т. д.). Дополнительная информация приведена в спецификации языка HTML. Значение этого атрибута представляет собой список имен категорий, разделенных запятыми.

7.12.24.2.3 headers : Set(String) «XMLIDREF»: в этом атрибуте указан список ячеек заголовка, содержание которых служит заголовком для данной ячейки. Значение этого атрибута представляет собой список ссылок на идентификаторы ID ячеек заголовка, разделенных пробелами; эти ячейки должны быть поименованы, используя атрибут id. Авторы обычно используют атрибут заголовков, чтобы не визуальные пользовательские агенты могли отобразить информацию заголовков ячеек данных (например, информация заголовков произносится перед тем, как представить содержание ячейки), но этот атрибут может также использоваться в сочетании с таблицами стилей. См. также описание атрибута scope.

Значение стереотипа XMLIDREF описано в А.2 приложения А.

7.12.24.2.4 scope : StrucDoc.CellScope: в этом атрибуте указан список ячеек данных, для которых данная ячейка служит заголовком. Он может использоваться вместо атрибута headers, особенно для простых таблиц.

Этот атрибут должен иметь значение только в ячейках заголовков.

Если ему присвоено значение, то оно должно браться из перечисления StrucDoc.CellScope, приведенного в таблице 35.

Таблица 35 — Перечисление StrucDoc.CellScope

Уровень	Код	Описание	Определение
1	row	Строка	Текущая ячейка служит заголовком для остальных ячеек строки, которая ее содержит
1	col	Столбец	Текущая ячейка служит заголовком для остальных ячеек столбца, который ее содержит
1	rowgroup	Группа строк	Текущая ячейка служит заголовком для остальных ячеек группы строк, которая ее содержит
1	colgroup	Группа столбцов	Текущая ячейка служит заголовком для остальных ячеек группы столбцов, которая ее содержит

По умолчанию используется значение col.

7.12.24.2.5 rowspan : Integer: число строк, которые охвачены данной ячейкой. По умолчанию полагается равным 1.

7.12.24.2.6 colspan : Integer: число столбцов, которые охвачены данной ячейкой. По умолчанию полагается равным 1.

7.12.24.3 Ассоциации

7.12.24.3.1 parts : StrucDoc.CMGeneral[0..* упорядочен] «Anonymous»: содержание сноски.

Значение стереотипа Anonymous описано в А.2 приложения А.

7.12.24.4 Инварианты:

- таблицы не могут быть непосредственно вложенными.

Определение инвариантов на языке OCL:

```
inv "нет вложенных таблиц": parts.forAll(t | t.table.oclIsUndefined)
```

7.12.25 Тип данных StrucDoc.TRow

7.12.25.1 Описание

Специализация типа данных StrucDoc.TableItem

Строка таблицы.

7.12.25.2 Ассоциации

7.12.25.2.1 parts : Sequence(StrucDoc.TRowPart) «Anonymous»: содержание строки.

Значение стереотипа Anonymous описано в А.2 приложения А.

7.12.25.3 Инварианты

- должна быть задана по меньшей мере одна часть строки.

Определение инвариантов на языке OCL:

```
inv "должен существовать по меньшей мере один элемент": parts->notEmpty
```

7.12.26 Тип данных StrucDoc.TRowPart

7.12.26.1 Описание

Стереотип: «Choice»

Модель содержания, позволяющая ячейки данных (td) или ячейки заголовка (th). Стереотип Choice означает, что ровно один из атрибутов должен иметь значение, все остальные должны быть пустыми.

7.12.26.2 Атрибуты

7.12.26.2.1 td: StrucDoc.TCell: ячейка таблицы.

7.12.26.2.2 th: StrucDoc.TCell: ячейка заголовка таблицы.

Примечание — Ячейки заголовка иногда отображаются другим стилем, например полужирным шрифтом, и могут повторяться после разрыва страницы.

7.12.27 Тип данных StrucDoc.TRowGroup

7.12.27.1 Описание

Специализация типа данных StrucDoc.TableItem.

Группа строк. Может использоваться, чтобы задать одинаковый стиль отображения группы строк.

7.12.27.2 Атрибуты

7.12.27.2.1 tr : Sequence(StrucDoc.Trow): строки группы.

7.12.27.3 Инварианты:

- должна быть задана по меньшей мере одна строка.

Определение инвариантов на языке OCL:

```
inv "должен существовать по меньшей мере один элемент": tr->notEmpty
```

7.12.28 Тип данных StrucDoc.ColItem

7.12.28.1 Описание

Абстрактный. Специализация типа данных StrucDoc.TableItem

Абстрактный предшественник, описывающий общие свойства столбцов и групп столбцов.

7.12.28.2 Атрибуты

7.12.28.2.1 span : Integer: число столбцов, на которые распространяется данное определение столбца. По умолчанию равно 1.

7.12.28.2.2 width : StrucDoc.Length: длина столбца.

7.12.29 Тип данных StrucDoc.Col

7.12.29.1 Описание

Специализация типа данных StrucDoc.ColItem

Задает одинаковый стиль каждой ячейке столбца.

7.12.30 Тип данных StrucDoc.ColGroup

7.12.30.1 Описание

Специализация T+

Задает одинаковый стиль каждой ячейке группы столбцов.

7.12.30.2 Ассоциации

7.12.30.2.1 col : StrucDoc.Col[0..* упорядочен]: столбцы данной группы.

7.12.30.3 Инварианты:

- должен быть задан по меньшей мере один столбец.

Определение инвариантов на языке OCL:

```
inv "должен существовать по меньшей мере один элемент": tr->notEmpty
```

7.12.31 Тип данных StrucDoc.Table

7.12.31.1 Описание

Специализация типа данных StrucDoc.Captioned

Таблица. Может иметь название и должна иметь по меньшей мере один столбец. Таблица может иметь необязательные строки заголовка и итогов. Все строки определены в группах. Таблица может также содержать элементы col и colgroup, задающие стили представления столбцов.

7.12.31.2 Атрибуты

7.12.31.2.1 summary : String: этот атрибут содержит краткое описание назначения и структуры таблицы, предназначенное для использования не визуальными пользовательскими агентами, например, речевыми или использующими письмо Брайля. Его отличие от названия в том, что он должен содержать неформатированный текст и обычно длиннее.

7.12.31.2.2 width: StrucDoc.Length: этот атрибут указывает желательную ширину всей таблицы. Он предназначен для визуальных пользовательских агентов.

Правила вычисления ширины таблицы, описанные в спецификации языка HTML, применимы и к таблицам, описанным в настоящем стандарте.

7.12.31.2.3 border : StrucDoc.Length: ширина границы.

7.12.31.2.4 frame : StrucDoc.Frame: этот атрибут указывает, какие стороны обрамления таблицы будут видимы (то есть, какие границы видимы).

Если этому атрибуту присвоено значение, то оно должно браться из перечисления StrucDoc.Frame, приведенного в таблице 36.

Таблица 36 — Перечисление StrucDoc.Frame

Уровень	Код	Описание	Определение
1	void	Отсутствует	Нет обрамления
1	above	Верх	Только верхняя сторона
1	below	Низ	Только нижняя сторона
1	hsides	Боковые стороны	Только правая и левая сторона
1	lhs	Левая сторона	Только левая сторона
1	rhs	Правая сторона	Только правая сторона
1	vsides	Верх и низ	Только верхняя и нижняя сторона
1	box	Прямоугольник	Все четыре стороны
1	border	Обрамление	Все четыре стороны

По умолчанию обрамления нет.

7.12.31.2.5 rules : StrucDoc.Rules: этот атрибут указывает, какие линии (то есть границы) появятся между ячейками таблицы. Правила отображения линий зависят от пользовательского агента.

Если этому атрибуту присвоено значение, то оно должно браться из перечисления StrucDoc.Rules, приведенного в таблице 37.

Таблица 37 — Перечисление StrucDoc.Rules

Уровень	Код	Описание	Определение
1	none	Отсутствуют	Линий нет
1	groups	Группы	Линии появятся только между группами строк и столбцом

Окончание таблицы 37

Уровень	Код	Описание	Определение
1	rows	Строки	Линии появятся только между строками
1	cols	Столбцы	Линии появятся только между столбцами
1	all	Все	Линии появятся только между всеми строками и столбцами

По умолчанию линий нет.

7.12.31.2.6 cellspacing . StrucDoc.Length: этот атрибут указывает, какое расстояние должен оставить пользовательский агент между левой стороной таблицы и левым краем самого левого столбца, верхней стороной таблицы и верхним краем самого верхнего столбца и так далее для правой и нижней стороны таблицы. Этот атрибут задает также расстояние между ячейками.

7.12.31.2.7 cellpadding : StrucDoc.Length: этот атрибут указывает, какое расстояние должно быть оставлено между краем ячейки и ее содержанием. Если значение этого атрибута задано в пикселах, то содержание ячейки должно находиться на этом расстоянии от всех четырех границ. Если значение этого атрибута задано в процентах, то верхняя и нижняя граница должны быть равноудалены от содержания на расстояние, равное заданному проценту от доступной высоты, а левая и правая граница должны быть равноудалены от содержания на расстояние, равное заданному проценту от доступной ширины.

Примечание — Дополнительная информация приведена в спецификации языка HTML.

7.12.31.3 Ассоциации

7.12.31.3.1 thead : TRowGroup: необязательная группа строк, представляющая собой заголовок таблицы.

7.12.31.3.2 tfoot : TRowGroup: необязательная группа строк, представляющая собой нижний колонтитул таблицы.

7.12.31.3.3 tbody : TrowGroup[0..* ordered]: необязательная группа строк, представляющая собой тело таблицы.

7.12.31.4 Инварианты:

- должна быть задана по меньшей мере одна строка.

Определения инвариантов на языке OCL:

```
inv "должна быть задана по меньшей мере одна строка": thead.tr->count
    + tfoot.tr->count + tbody.tr->count > 0
```

7.12.32 Тип данных SD.TEXT

7.12.32.1 Описание

Специализация типа данных ANY

Определение структурированного текста, который может быть использован в здравоохранении.

Структурированный текст основан на принципах, подобных языку XHTML, которые обеспечивают разметку текста в соответствии с необходимой семантикой и служат основой для реализации структурированного текста в здравоохранении.

Тип данных SD.TEXT известен также под унаследованным именем StrucDoc.Text.

7.12.32.2 Ассоциации

7.12.32.2.1 base : StrucDoc.Base[1..1] «Anonymous»: атрибуты базовой идентификации и стилей представления.

7.12.32.2.2 parts : StrucDoc.CMGeneral[0..* ordered] «Anonymous»: содержание структурированного текста.

Значение стереотипа Anonymous описано в А.2 приложения А.

7.12.32.3 Равенство

Два значения типа SD.TEXT равны, если они не имеют причины пустоты nullFlavor и имеют равное содержание (атрибуты language, styleCode и parts).

7.12.32.4 Инварианты:

- непустой экземпляр типа данных SD.TEXT должен иметь некоторые части parts.

Определение инвариантов на языке OCL:

```
inv "если нет причины пустоты, должно иметься значение":
    isNotNull implies parts >count > 0
```

7.12.32.5 Операции

7.12.32.5.1 asED() : ED: представление значения типа SD.TEXT в качестве значения типа ED. Атрибут parts становится значением атрибута ED.xml в соответствии с правилами преобразования в формат XML, изложенными в приложении А. Атрибуту language присваивается значение атрибута base. language, атрибуту mediaType — значение «text/x-hl7-text+xml». Значение атрибута charset определяется контекстом, в котором используется значение типа SD.TEXT, а остальным свойствам значения типа ED присваиваются пустые значения.

7.12.33 Тип данных SD.TITLE

7.12.33.1 Описание

Специализация типа данных ANY

Определение структурированного заголовка, который может быть использован в здравоохранении.

Используется то же определение структурированного текста, что и для типа данных SD.TEXT, но при этом набор свойств более ограничен, поскольку он присваивается заголовку, а не документу общего вида.

Тип данных SD.TITLE известен также под унаследованным именем StrucDoc.Title.

7.12.33.2 Ассоциации

7.12.33.2.1 base . StrucDoc.Base[1..1] «Anonymous»: атрибуты базовой идентификации и стилей представления.

7.12.33.2.2 parts : StrucDoc.CMTitle[0..* ordered] «Anonymous»: содержание заголовка.

Значение стереотипа Anonymous описано в А.2 приложения А.

7.12.33.3 Равенство

Два значения типа SD.TITLE равны, если они не имеют причины пустоты nullFlavor и имеют равное содержание (атрибуты language, styleCode и parts).

7.12.33.4 Инварианты:

- непустой экземпляр типа данных SD.TITLE должен иметь некоторые части parts.

Определение инвариантов на языке OCL:

```
inv "если нет причины пустоты, должно иметься значение":
    isNotNull implies parts->count > 0
```

7.12.33.5 Операции

7.12.33.5.1 asED() : ED: представление значения типа SD.TITLE в качестве значения типа ED. Атрибут parts становится значением атрибута ED.xml в соответствии с правилами преобразования в формат XML, изложенными в приложении А. Атрибуту language присваивается значение атрибута base. language, атрибуту mediaType — значение «text/x-hl7-title+xml». Значение атрибута charset определяется контекстом, в котором используется значение типа SD.TITLE, а остальным свойствам значения типа ED присваиваются пустые значения.

Приложение А
(обязательное)

Представление на языке XML

А.1 Введение

В связи с повсеместным использования языка XML в качестве формата обмена в настоящем стандарте описан нормативный формат представления экземпляров определенных в нем типов данных на языке XML. В объявлении соответствия элемента обработки информации настоящему стандарту должно быть указано, принят ли этот формат и если да, то в какой мере.

Представление на языке XML создается с помощью простого алгоритма, описанного ниже. Оно предназначено для обеспечения возможности проверки соответствия XML-схеме и для использования в средствах разработки программного обеспечения, основанных на использовании XML-схем. Полная XML-схема представления описана ниже, в справочном приложении. Хотя эта схема является справочной, элемент обработки информации, объявляющий соответствие настоящему стандарту, должен создавать экземпляры типов данных, соответствующих этой схеме, хотя этого и недостаточно для объявления, что соответствие продемонстрировано.

А.2 Правила представления на языке XML

Должны быть выполнены следующие правила представления на языке XML:

a) может быть использован любой набор символов, допустимый в языке XML, при условии совместимости с набором символов, используемым в типе данных String (см. 6.7.5). В частности, требуется взаимно-однозначное соответствие этих двух наборов символов;

b) атрибут xml:lang должен игнорироваться. В содержании, к которому применим язык, вместо него должны использоваться атрибуты Data.language или ST.language;

c) все элементы должны принадлежать некоторому пространству имен, которое должно быть определено в объявлениях соответствия элементов обработки информации настоящему стандарту. Пространство имен «uri:iso.org:21090» зарезервировано для прямого применения типов данных, например, в целях тестирования;

d) представление на языке XML (и XML-схема) типа данных производится алгоритмически от представления на языке UML;

e) каждый тип данных, являющийся специализацией типа данных ANY, представляется XML-элементом;

f) атрибуты UML, тип которых имеет стереотип Binary, представляются элементом с текстовым содержанием:

1) формат текста в этом элементе представляет собой формат base64Binary, описанный в спецификации XML-схемы, разработанной консорциумом W3C;

g) атрибуты UML, тип которых имеет стереотип XML, следующим образом представляются на языке XML в виде одного элемента:

1) имя атрибута UML служит именем XML-элемента;

2) значение this является эквивалентом типа anyType в XML-схеме;

h) атрибуты UML, имеющие стереотип XMLID, представляются на языке XML как атрибут типа ID;

i) атрибуты UML, имеющие стереотип XMLIDREF, представляются на языке XML как атрибут типа IDREF, имя которого совпадает с именем атрибута UML. Если типом атрибута UML является коллекция, то на языке XML значение атрибута представляет собой список токенов IDREF, разделенных пробелами;

j) атрибуты UML, имеющие стереотип XMLText, представляются на языке XML как текстовое содержание. Каждый класс может иметь не более одного атрибута с этим стереотипом;

k) атрибуты UML, тип которых является производным от класса Classifier, следующим образом представляются как элемент:

1) имя XML-элемента наследуется от имени атрибута UML;

2) формат элемента тот же, что у этого типа, при рекурсивном выполнении настоящих правил;

l) атрибуты UML, тип которых является производным от примитивного класса UML, представляются как XML-атрибут следующим образом:

1) имя XML-элемента наследуется от имени атрибута UML;

2) атрибут не имеет пространства имен;

3) содержание атрибута должно иметь тип из числа определенных консорциумом W3C для XML-схем, задаваемый в соответствии с таблицей А.1;

4) если значение атрибута равно его значению по умолчанию, то он не обязан присутствовать в XML-представлении; значение атрибута по умолчанию является пустым, если иное не указано на UML-диаграмме;

m) атрибуты UML, тип которых является производным от коллекций классов UML, следующим образом представляются на языке XML как последовательность XML-элементов:

1) имя XML-элемента наследуется от имени атрибута UML;

2) каждому элементу коллекции соответствует один элемент;

3) формат элемента тот же, что у типа элемента коллекции, при рекурсивном выполнении настоящих правил;

n) атрибуты UML, тип которых является производным от коллекций примитивных классов UML, представляются как XML-атрибут следующим образом:

1) имя XML-элемента наследуется от имени атрибута UML;

2) атрибут не имеет пространства имен;

3) содержание атрибута должно быть списком токенов, разделенных пробелами. Формат токена должен соответствовать типу, определенному в схеме и совпадающему с типом UML в соответствии с таблицей А.1;

4) если значение атрибута равно его значению по умолчанию, то он не обязан присутствовать в XML-представлении; значение атрибута по умолчанию является пустым, если иное не указано на UML-диаграмме;

o) UML-ассоциации со стереотипом «Anonymous» вообще не представляются каким-либо элементом;

p) атрибут xsi:nil не должен использоваться в представлении значения на языке XML;

q) если тип, подлежащий представлению, отличается от того, который описан на языке UML (например, является специализацией), то у элемента должен быть задан атрибут xsi:type. Этот атрибут может использоваться в любое время;

r) имя типа данных в схеме должно совпадать с именем класса UML, за исключением связанных параметризованных классов;

s) для связанных параметризованных классов (имеющих в качестве параметров типы данных) имена фактических параметров добавляются к имени класса, используя в качестве разделителя знак подчеркивания «_», например, имя DSET(AD) превращается в DSET_AD;

t) если атрибут содержится по ссылке, то он представляется, используя XML-атрибуты ID/IDREF. Цель ссылки должна находиться в том же самом документе. Сам атрибут может иметь другие ограничения того, как может быть найдена цель ссылки. Ссылки должны быть прямыми.

А.3 Представление тонкостей типов данных на языке XML

Тонкости типов данных используются для указания того, что на тип данных наложено одно или несколько множеств ограничений. Тонкости могут быть применены к экземпляру типа данных непосредственно, используя свойство flavorId, или опосредованно, используя какие-либо иные знания. Однако применяемые тонкости не трактуются как типы на языке XML, поскольку к одному типу может применяться несколько тонкостей, а в XML-схемах нет возможности задавать такие ограничения.

Вместо этого ограничения, ассоциированные с каждой тонкостью, определенной в настоящем стандарте, описаны на языке правил Schematron, содержание которых может быть представлено на языке XPath и в таком виде включено в XML-схему. Хотя не требуется, чтобы тонкости, не определенные в настоящем стандарте, были описаны на языке Schematron, другие элементы обработки информации могут воспользоваться образцами, включенными в XML-схему, при описании иных тонкостей типов данных.

А.4 Отображение типов данных UML в XML-схеме

Отображение примитивных типов UML в XML-схеме приведено в таблице А.1.

Таблица А.1 — Отображение примитивных типов UML в XML-схеме

Примитивный тип UML	Тип данных в XML-схеме
Boolean	boolean
Integer	integer
Decimal	decimal
String	string

Если тип данных UML имеет стереотип Uri, то вместо строкового типа используется тип anyURI, определенный консорциумом W3C в XML-схемах.

А.5 XML-схема

На основе описанных выше правил отображения можно сконструировать XML-схему. Копия одной из таких схем приведена в приложении Е. Эта схема может использоваться для проверки того, соответствуют ли экземпляры типов данных, представленные на языке XML, некоторым из правил, изложенных в настоящем стандарте.

В схему включены объявления на языке Schematron, представляющие некоторые из ограничений, описанных в настоящем стандарте на языке OCL.

Схема не может обеспечить полную проверку соответствия. Факт, что какой-либо конкретный экземпляр типа данных соответствует схеме, установленный с помощью некоторого средства проверки соответствия, не доказывает, что этот экземпляр соответствует настоящему стандарту. Чтобы убедиться в полном соответствии, приложения должны предпринимать дополнительные проверки данных на соответствие правилам целостности, не обеспечиваемые XML-схемой. Если же экземпляр типа данных не прошел проверку на соответствие схеме, и это не вызвано ошибкой или отсутствием некоторой функции в используемом средстве проверки, то этот экземпляр не соответствует настоящему стандарту.

При импорте какой-либо схемы необходимо обеспечить присвоение пространства имен типам данных, описанным в этой схеме. необходимо учитывать, что ни один из элементов, которым присвоены типы, описанные в этой схеме, не может иметь атрибут xsi:nil.

Приложение В
(обязательное)

Вспомогательные типы данных UML

В.1 Общие положения

В данном разделе определены некоторые типы данных UML, необходимые для поддержки объявлений описанных выше типов UML.

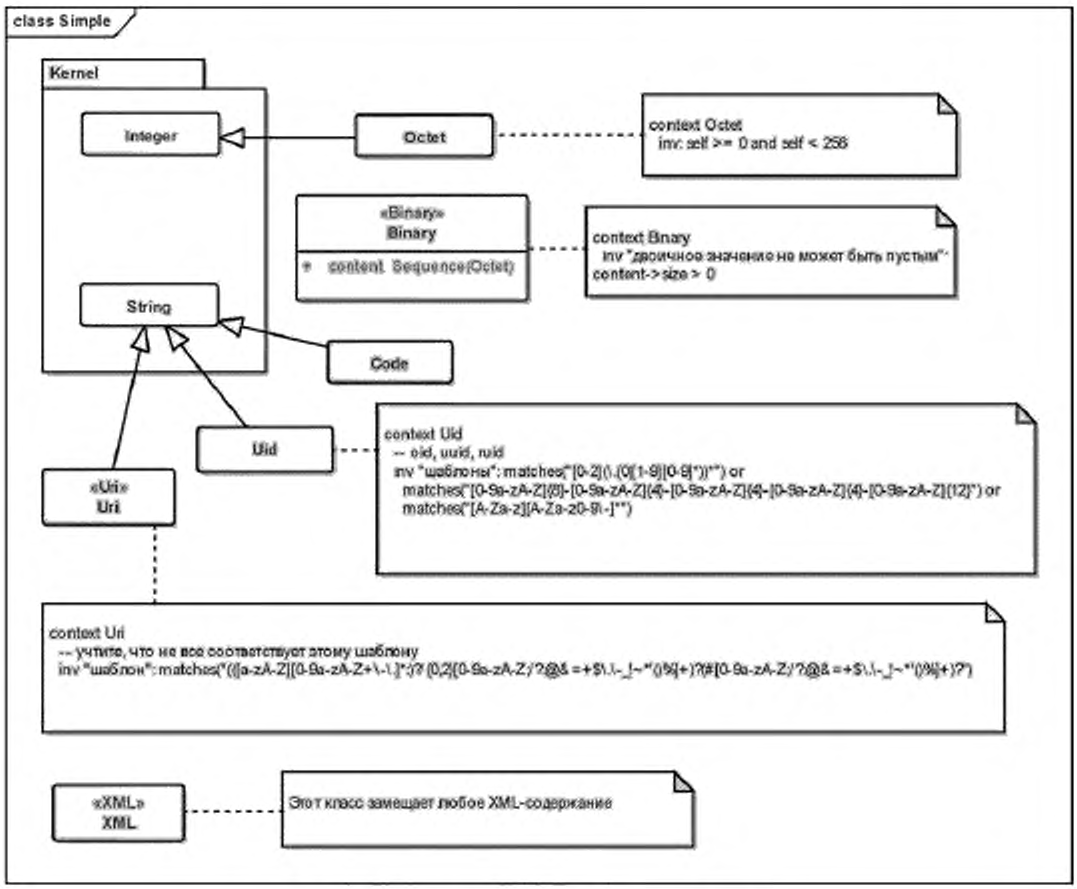


Рисунок В.1 — Вспомогательные типы данных UML

В.2 Типы данных UML

В.2.1 Тип данных Octet

Представляет число, которое может принимать значение в диапазоне 0—255 (известное также как байт). Этот тип данных определен, чтобы можно было дать формальное определение типа данных Binary, используемого в языке UML.

B.2.2 Тип данных Binary

Представляет последовательность значений типа Octet. Этот тип данных специально сконструирован, чтобы в объявлениях на языке UML можно было задавать двоичное содержание (иногда называемое потоком).

Этот тип имеет стереотип <<Binary>>, позволяющий обеспечивать автоматическое отображение для специфичных платформ, в которых обычно предусмотрены конкретные типы данных, позволяющие манипулировать двоичным содержанием.

B.2.3 Тип данных Code

Простая строка, имеющая ограниченный набор возможных значений (кодов). Такие коды обычно используется, когда список значений ограничен каким-либо внешним стандартом, например, спецификацией RFC.

B.2.4 Тип данных Uid

Строка, которая служит идентификатором и, следовательно, может представлять собой строковое представление объектного идентификатора [ОИД, см. ИСО/МЭК 8824:1990(E)], универсальный уникальный идентификатор (UUID, см. приложение A к спецификации CDE 1.1 Remote Procedure Call, принятой организацией Open Group) или простой токен, взятый из списка контролируемых имен, предназначенного для контекста, в котором используются значения типа данных Uid.

Значения типа данных Uid всегда должны представляться в верхнем регистре, и сравнение значение типа Uid всегда чувствительно к регистру.

B.2.5 Тип данных Uri

Простой адрес URL или URI, используемый в Интернете. Он представляет собой расширение строкового типа, но имеет стереотип <<URI>>, позволяющий обеспечивать автоматическое отображение для специфичных платформ, в которых обычно предусмотрены конкретные типы данных, позволяющие манипулировать двоичным содержанием.

B.2.6 Тип данных XML

Замещает любое XML-содержание. Он используется в классе Data, чтобы описывать или двоичное содержание, или объектную модель на языке XML.

B.2.7 Тип данных Decimal

Число с плавающей точкой, имеющее известную точность. Операции над числами с плавающей точкой должны учитывать их точность. Этот тип данных не отображается на простые типы real, определенные в ядре языка UML или в ИСО/МЭК 11404, поскольку в них не учитывается точность.

Примечания

1 Одним из применений типа данных Decimal в настоящем стандарте является представление денежных сумм, к которому предъявляются хорошо известные требования точности.

2 Свойство точности относится только к десятичному представлению числа, а не к точности вещественного значения. Это понятие предназначено для верной передачи человеку всей информации, представленной числом. Количество показанных десятичных цифр передает информацию о неопределенности (то есть точности и погрешности) измеренного значения.

Действуют следующие правила определения значащих цифр:

1) все ненулевые цифры являются значащими;

2) все нули справа от значащей цифры являются значащими;

3) если все цифры числа являются нулями, то нуль, находящийся непосредственно слева от десятичной точки, является значащим (и в силу правила 2 все следующие нули также являются значащими).

Примечание — Эти правила определения значащих цифр несколько отличаются от более привычных правил, преподаваемых в школе, а именно здесь все концевые нули перед десятичной точкой последовательно считаются значащими. Иначе, например, в числе 2000 нельзя определить, какие нули являются значащими. Это отклонение от привычного правила предназначено для обеспечения однозначной коммуникации.

Примеры точности литералов вещественных чисел приведены в таблице B.1.

Таблица B.1 — Примеры точности литералов вещественных чисел

Литерал	Число значащих цифр
2000	Четыре значащие цифры
2e3	Одна значащая цифра, само число могло бы быть записано как «2000», но точность составляет только один десятичный разряд
0.001	Одна значащая цифра
1e-3	Одна значащая цифра, само число могло бы быть записано как «0.001», но точность составляет только один десятичный разряд

Окончание таблицы В.1

Литерал	Число значащих цифр
0	Одна значащая цифра
0.0	Две значащие цифры
000.0	Две значащие цифры
0.00	Три значащие цифры
4.10	Три значащие цифры
4.09	Три значащие цифры
4.1	Две значащие цифры

Точность представления значения не зависит от его неопределенности (погрешности). Если важно иметь информацию о неопределенности значения, то для его описания надо использовать типы данных PPD или CIVL.

Точность представления должна совпадать с неопределенностью значения. Однако точность представления и неопределенность значения представляют собой два независимых понятия. Детальное обсуждение неопределенности вещественных значений приведено в описании типа данных PPD<REAL>.

Например, представление «0.123» имеет три значащие цифры, но неопределенность значения может быть в любом показанном или не показанном десятичном разряде, например, неопределенность может быть 0,123 ± 0,0005, 0,123 ± 0,005 или 0,123 ± 0,00005 и т. д. Но поскольку точность строки цифр кратна 0.5 от наименьшего значащего разряда, то неопределенность принимать любое значение между этими «пределами» и 0,123 ± 0,005 будет также адекватным представлением значения, находящегося в диапазоне от 0,118 до 0,128.

Приложение С
(справочное)

Отображение на точки зрения БМ-ОРО

Базовая модель открытой распределенной обработки БМ-ОРО (Reference Model of Open Distributed Processing — RM-ODP) совместно разработана организациями ИСО, МЭК и ITU-T. Она предназначена для координации стандартизации открытой распределенной обработки (ОРО). Семейство рекомендаций и международных стандартов БМ-ОРО определяет существенные понятия, необходимые для описания систем открытой распределенной обработки с пяти предписанных точек зрения, и обеспечивает хорошо разработанную базу для структурирования спецификация масштабных распределенных систем.

БМ-ОРО определяет следующие пять точек зрения на описание системы:

- предпринимательская точка зрения, которая сконцентрирована на целях сфере применения и политике, управляющей деятельностью специфицируемой системы в организации, частью которой она является;
- информационная точка зрения, которая сконцентрирована на видах информации, обрабатываемой системой, и ограничениях на использование и интерпретацию этой информации;
- вычислительная точка зрения, которая сконцентрирована на функциональной декомпозиции системы на множество объектов, взаимодействующих через интерфейсы: такая декомпозиция нужна для распределения системы;
- инженерная точка зрения, которая сконцентрирована на инфраструктуре, требуемой для поддержки распределения системы;
- технологическая точка зрения, которая сконцентрирована на выборе технологии для поддержки распределения системы.

Хотя между любой из этих точек зрения и настоящим стандартом типов данных нет прямого отображения, тем не менее они имеют определенную концептуальную близость.

Спецификация абстрактных типов данных HL7 V3 Abstract Data Types концептуально близка информационной точке зрения. Она сконцентрирована на видах информации, которые могут обрабатываться, и ограничениях, накладываемых на использование и интерпретацию информации. В ней нет прямых ответов на вычислительные аспекты, возникающие при фактическом конструировании систем.

Настоящий стандарт концептуально близок вычислительной точке зрения: он сконцентрирован на том, как представить информационную точку зрения, описанную в документе HL7 V3 Abstract Data Types, для множества объектов и как решать такие вычислительные вопросы, как оперирование пустыми объектами и взаимодействие тонкостей типов данных.

На базе настоящего стандарта могут быть разработаны инструментальные средства, которые могут на основе спецификации, описанной на языке UML, создавать реализации, связанные с конкретной технологией, например, генерировать код на конкретном языке программирования. Такие генерируемые спецификации связаны с технологической точкой зрения и инженерной точкой зрения, а реализуемые ими интерфейсы — с вычислительной точкой зрения.

Приложение D
(справочное)

Отображение на абстрактные типы данных документа HL7 V3

Спецификация HL7 V3 Abstract Data Types Specification (R2) описывает типы данных, используемые в документа HL7 V3, в чисто семантической манере. Эти типы данных определяются независимо от любой другой спецификации подобно тому, как ИСО/МЭК 11404 определяет типы данных общего назначения.

Примечание — Сочетание настоящего стандарта со спецификацией Abstract Data Types вводит некоторые изменения типов данных, определенных в первом выпуске документа HL7 V3, которые не являются обратно совместимыми. Этот вопрос детально обсуждается в объявлении соответствия настоящему стандарту, приведенному в документе HL7 V3.

Настоящий стандарт представляет собой реализацию документа HL7 V3 Abstract Data Types (R2). На рисунке D.1 показана диаграмма классов UML, представляющая сводку абстрактных типов данных, определенных в этой спецификации.

Примечание — На рисунке D.1 абстрактные типы данных представлены как интерфейсы с пользовательским стереотипом, называемым «mixin». Интерфейс с этим стереотипом похож на параметризованный интерфейс, но вместо представления свойств типа параметра такой интерфейс расширяет свойства интерфейса, являющегося его параметром. Этот метод не поддерживается во многих технологиях реализации и не имеет непосредственной поддержки в языке UML.

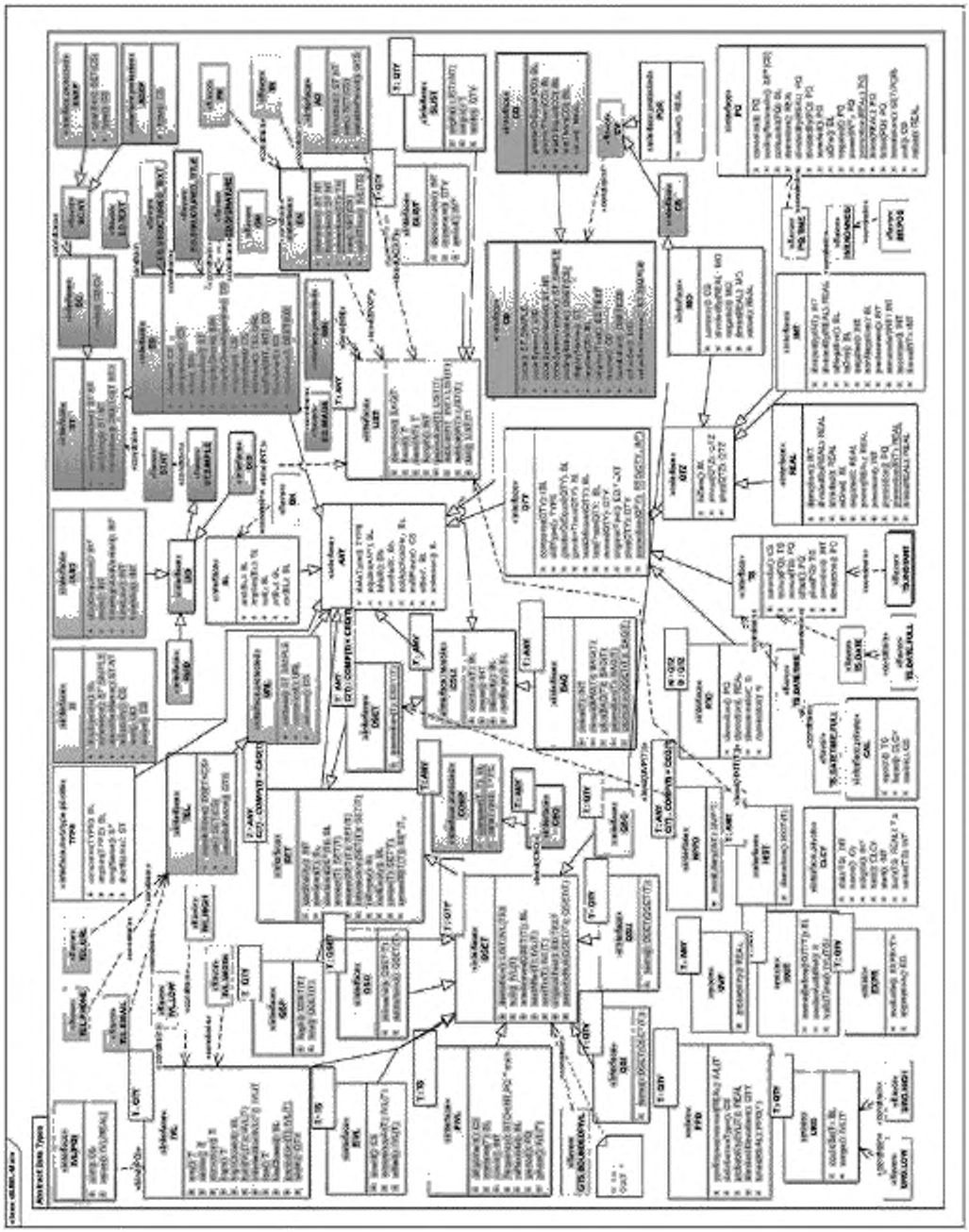


Рисунок D.1 — Сводка абстрактных типов данных

На рисунках D.2—D.11 показано, как типы данных, определенные в настоящем стандарте, реализуют абстрактные типы данных, описанные в спецификации HL7 V3 Abstract Data Types.

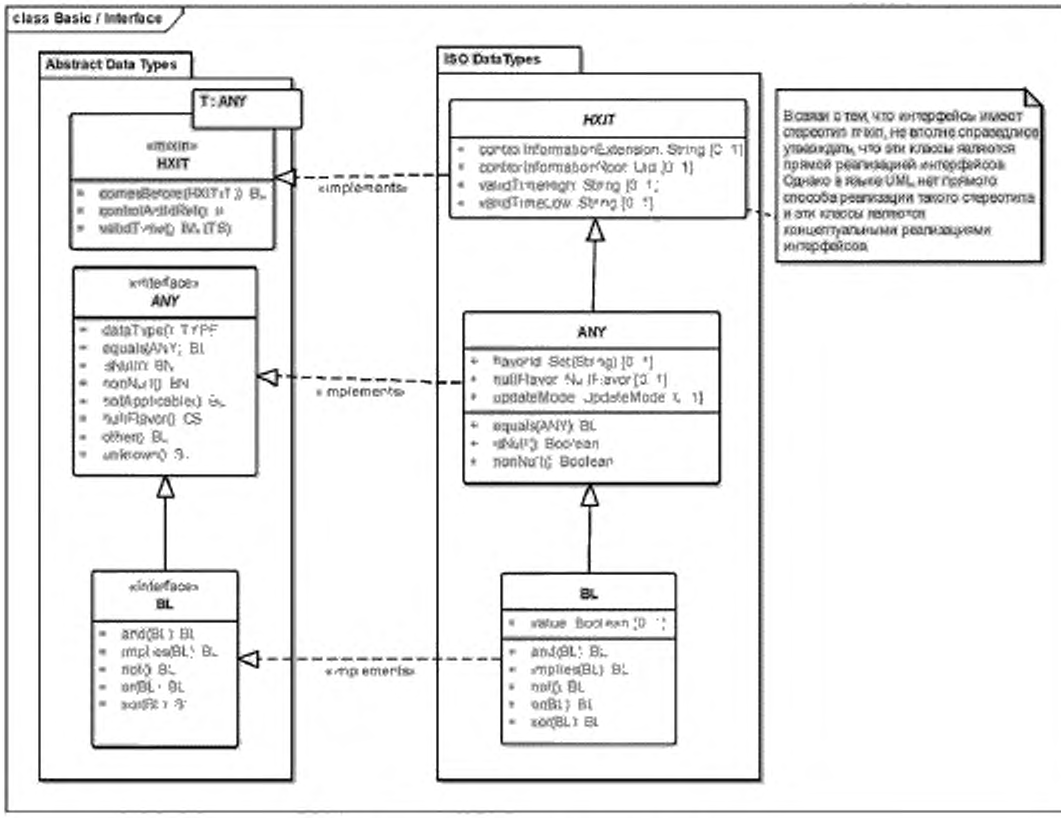


Рисунок D.2 — Отображение базовых типов данных

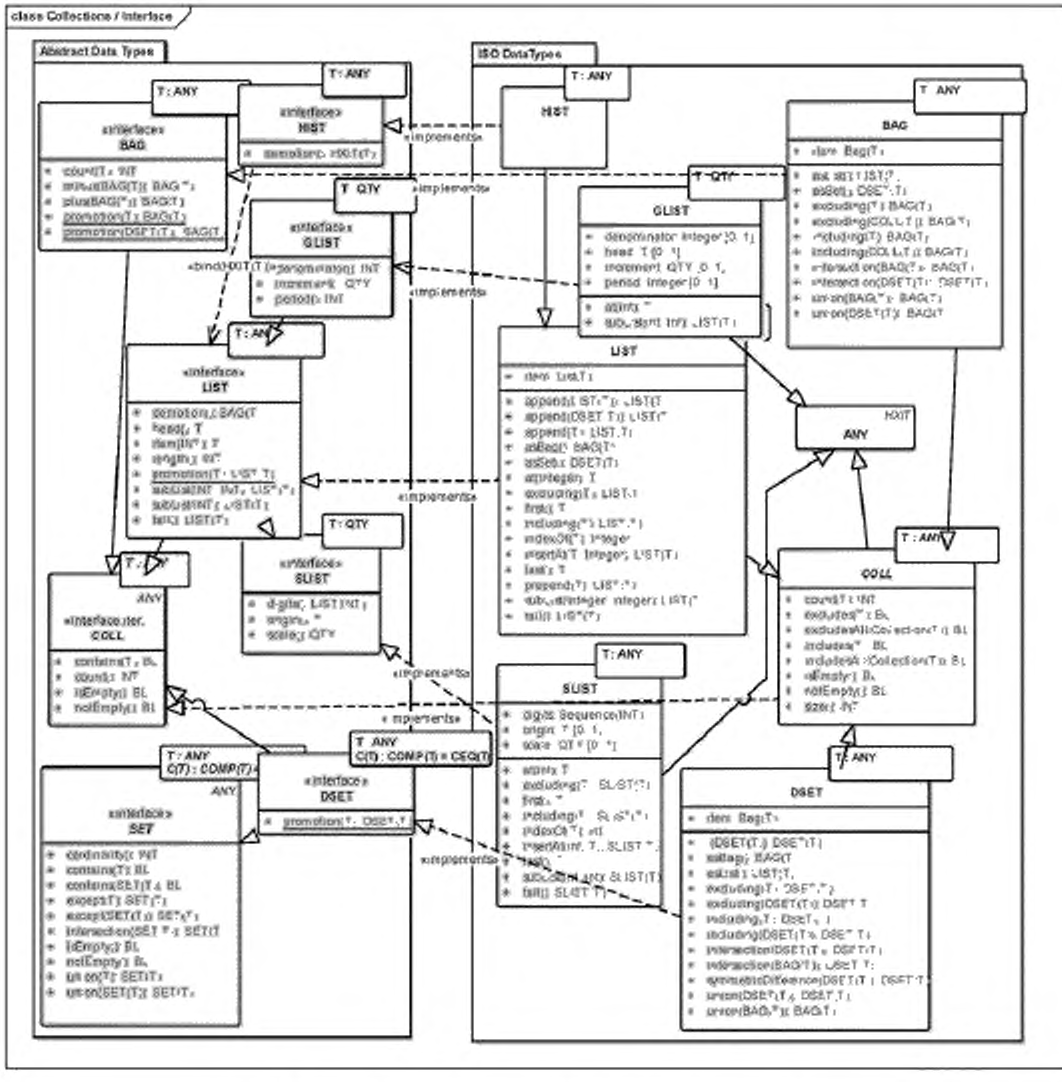


Рисунок D.3 — Отображение типов данных коллекций

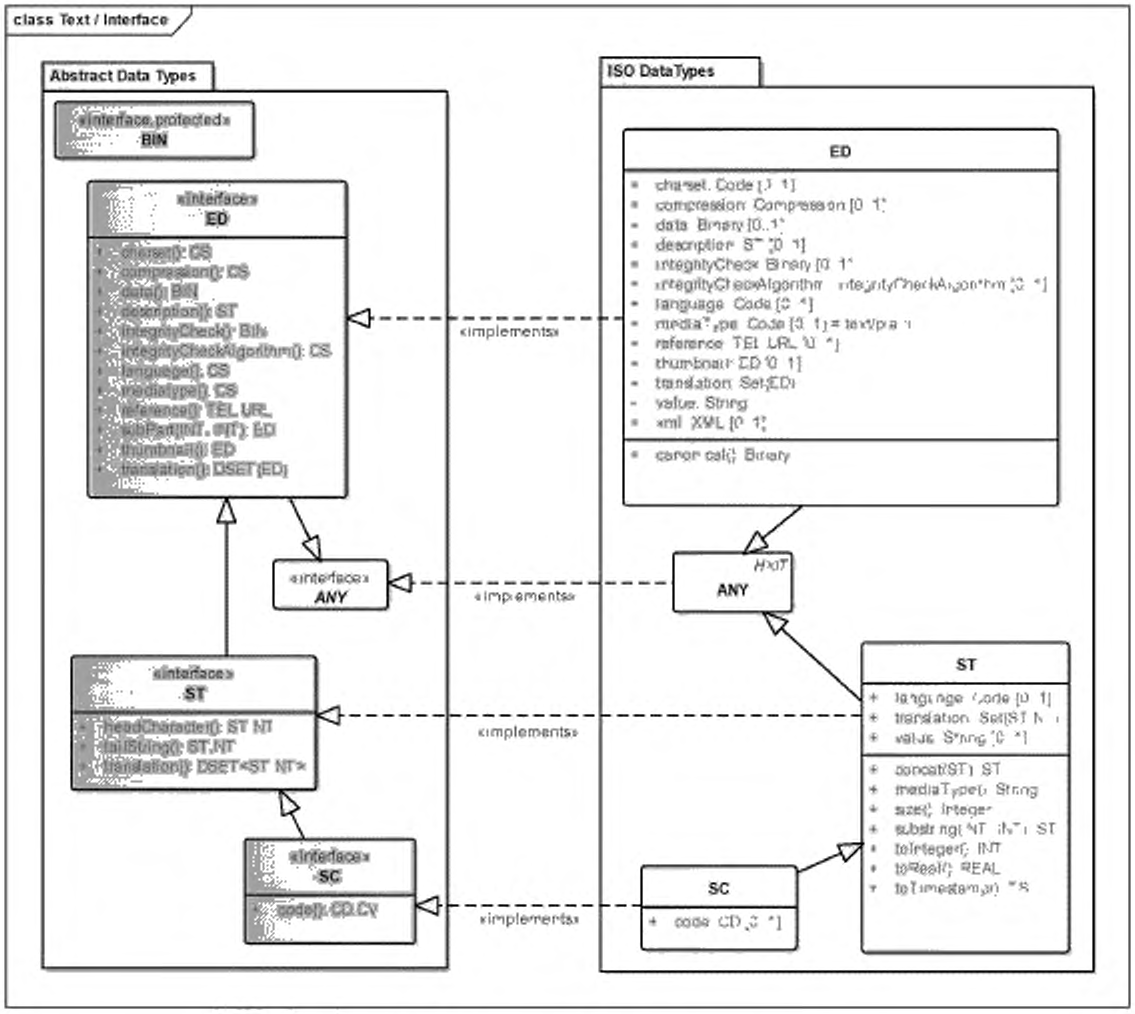


Рисунок D.4 — Отображение текстовых типов данных

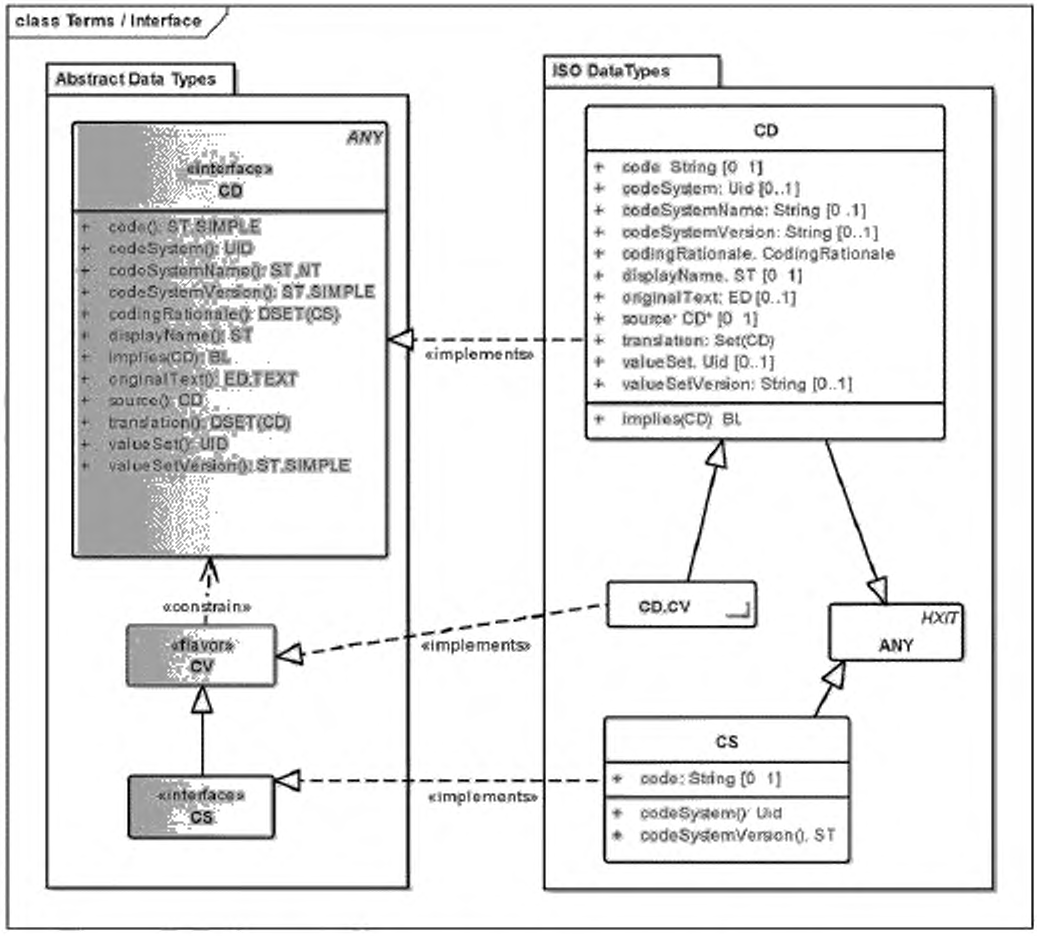


Рисунок D.5 — Отображение терминологических типов данных

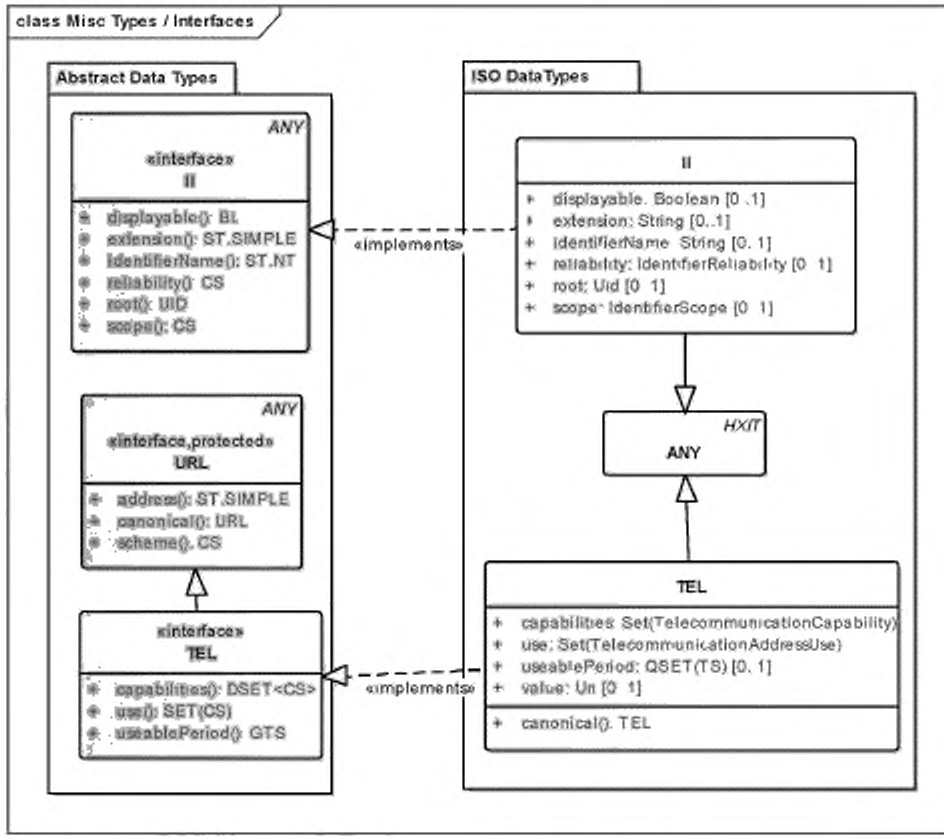


Рисунок D.6 — Отображение типов данных идентификации и местонахождения

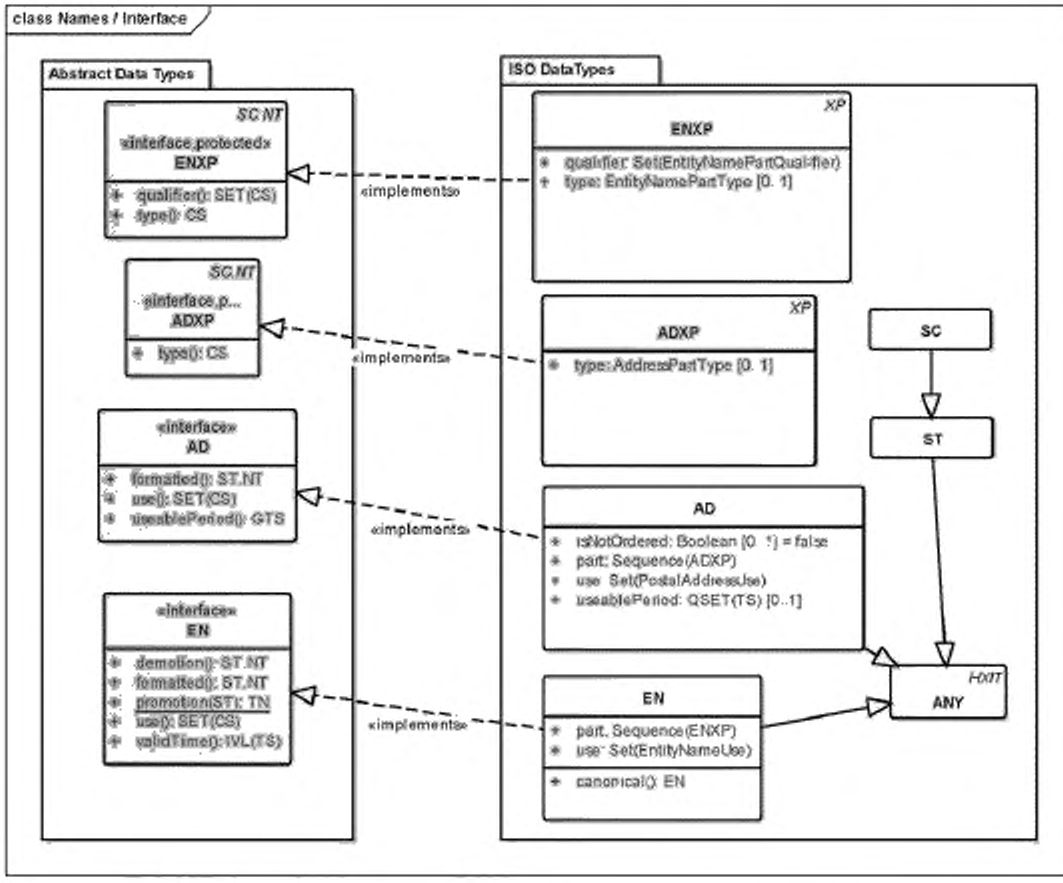


Рисунок D.7 — Отображение типов данных именования и адреса

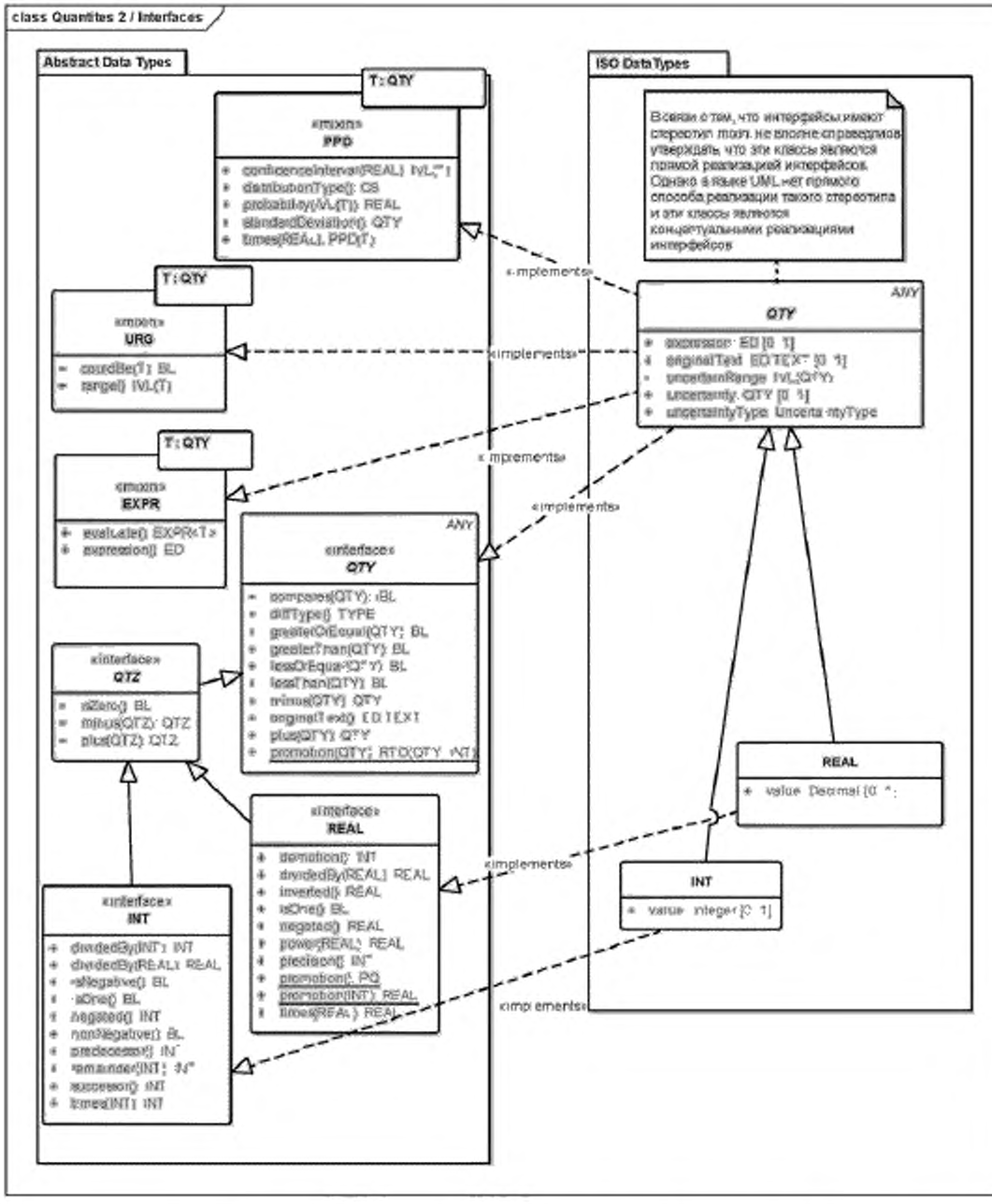


Рисунок D.8 — Отображение базовых количественных типов данных

Примечание — В типе данных QTY атрибут expression унаследован от типа данных EXPR<T>, атрибуты uncertainty и uncertaintyType унаследованы от типа данных PPD<T>, а атрибут uncertainRange — от типа данных URG<T>.

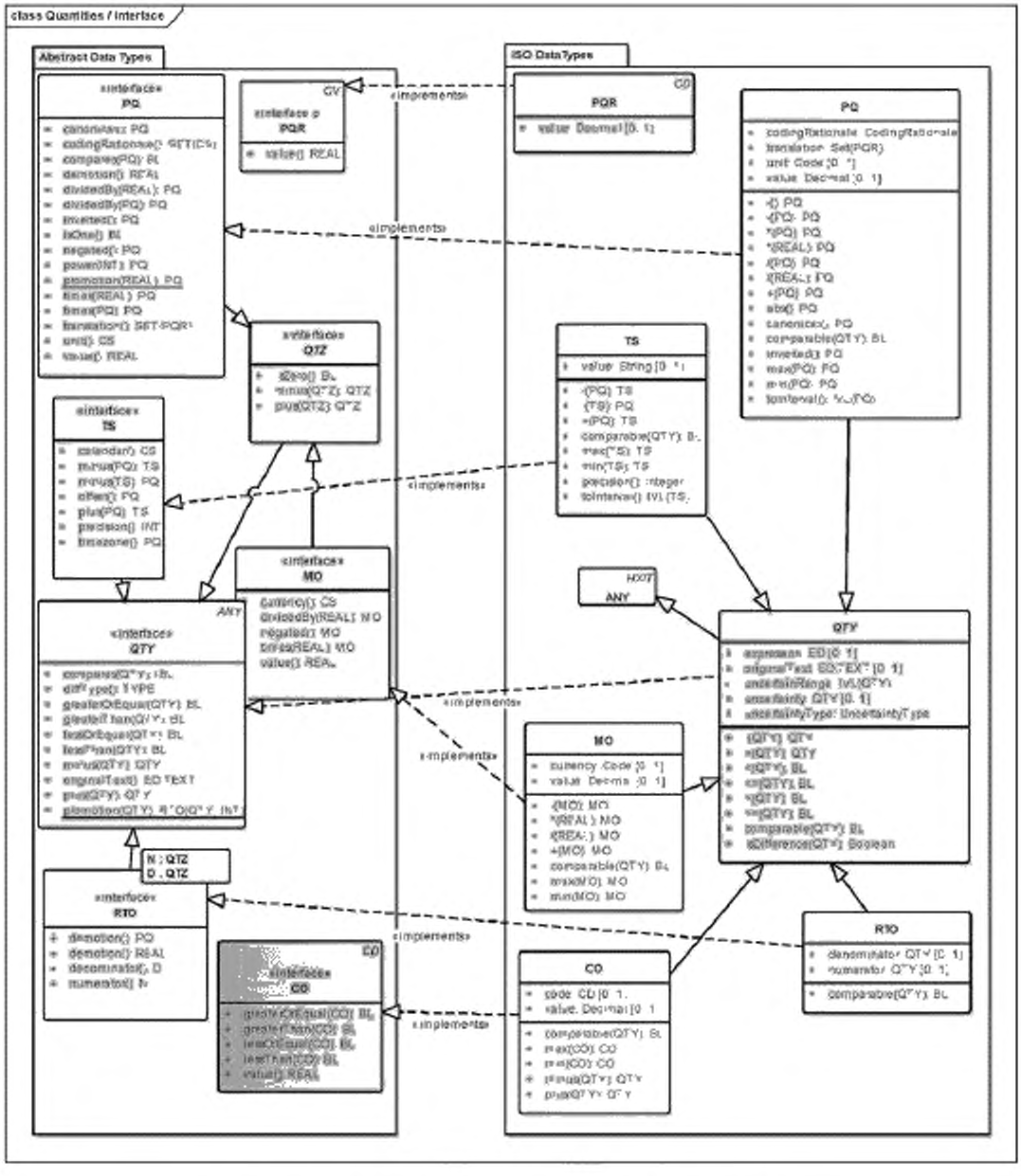


Рисунок D.9 — Отображение количественных типов данных

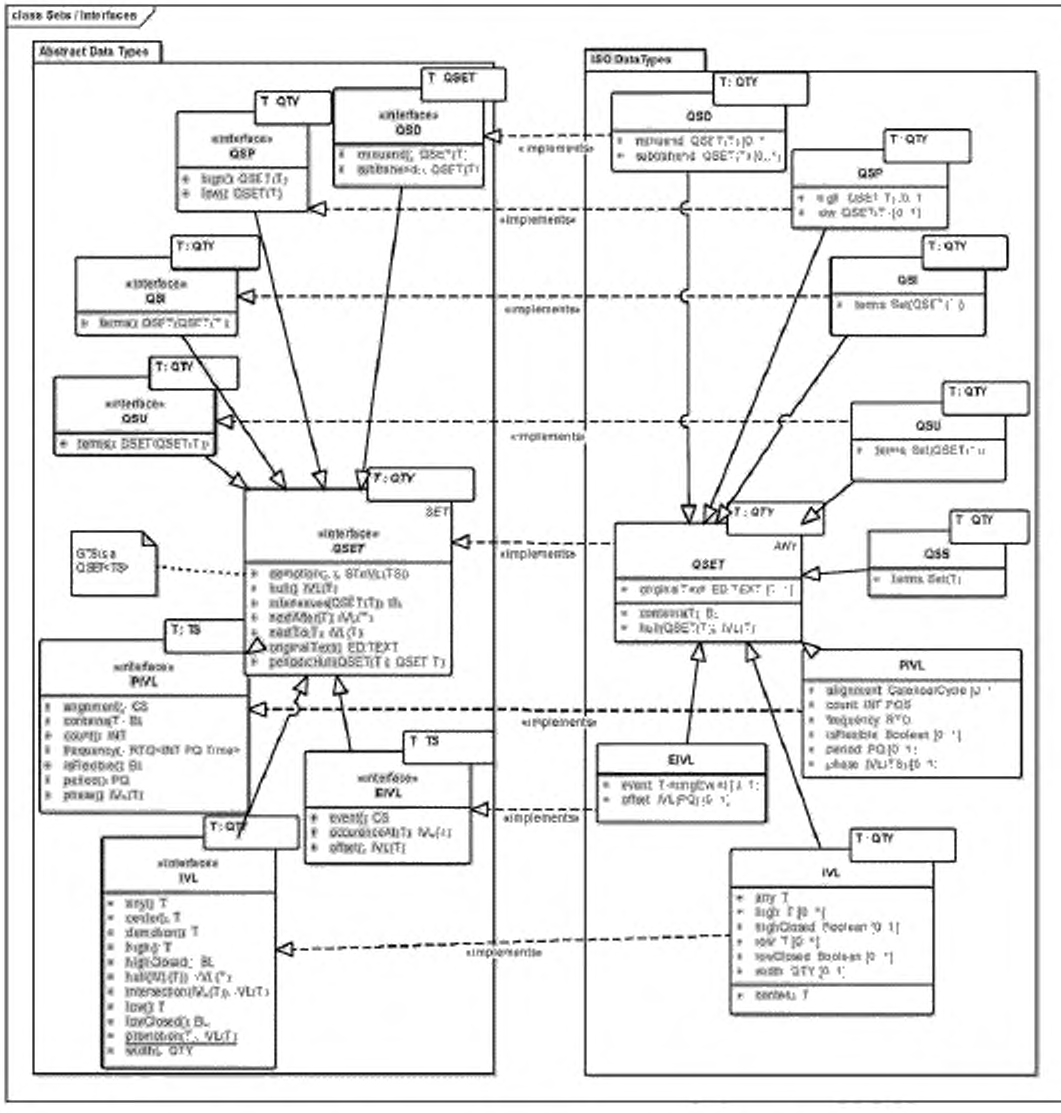


Рисунок D.10 — Отображение множеств количественных типов данных

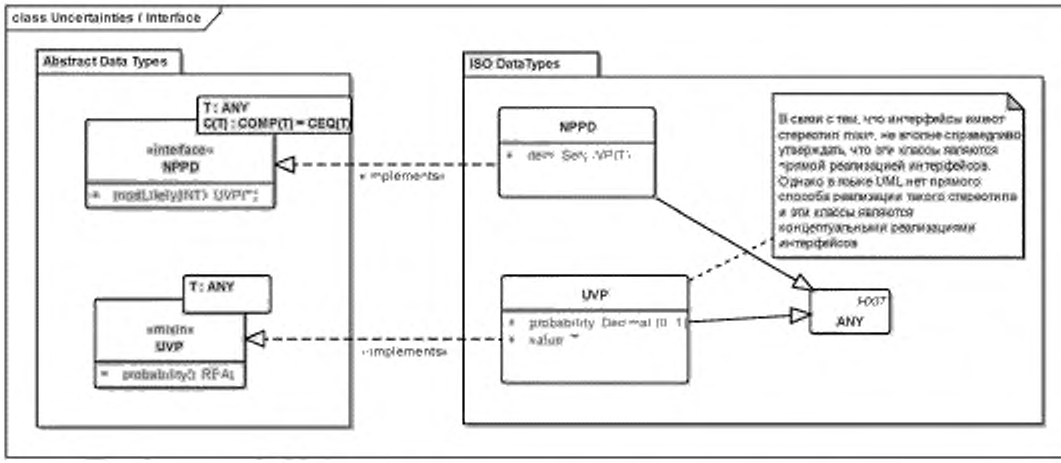


Рисунок D.11 — Отображение типов данных неопределенности

Приложение Е
(справочное)

Схема для представления на языке XML

XML-схема представления типов данных, описанных в настоящем стандарте, подготовлена в соответствии с правилами, описанными в разделе А.5. Ее можно загрузить со страницы gost.e-ecolog.ru permalink/?ISODataTypes2010.

Приложение ДА
(справочное)

Сведения о соответствии ссылочных международных стандартов и документов национальным и межгосударственным стандартам Российской Федерации

Таблица ДА.1

Обозначение ссылочного международного стандарта	Степень соответствия	Обозначение и наименование соответствующего национального и межгосударственного стандарта, документа
ИСО 4217	—	ОК (МК (ИСО 4217) 003-97) 014—2000 «Общероссийский классификатор валют»
ИСО/МЭК 8601	IDT	ГОСТ ИСО 8601—2001 «Система стандартов по информации, библиотечному и издательскому делу. Представление дат и времени. Общие требования»
ИСО/МЭК 8824:1990	IDT	ГОСТ Р ИСО/МЭК 8824—93 «Информационная технология. Взаимосвязь открытых систем. Абстрактно-синтаксическая нотация версии 1 (АСН.1)»
ИСО/МЭК 11404:2007	—	*
ISO/TS 22220	IDT	ГОСТ ISO/TS 22220—2013 «Информатизация здоровья. Идентификация субъектов медицинской помощи»
IETF RFC 1738	—	*
IETF RFC 1950	—	*
IETF RFC 1951	—	*
IETF RFC 1952	—	*
IETF RFC 2045	—	*
IETF RFC 2046	—	*
IETF RFC 2396	—	*
IETF RFC 2806	—	*
IETF RFC 3066	—	*
FIPS PUB 180-1	—	*
FIPS PUB 180-2	—	*
Open Group, CDE 1.1	—	*
HL7 V3 Standard, Data Types — Abstract Specification (R2)	—	—
Regenstrief Institute, Inc. and the UCUM Organization The Unified Code for Units of Measure	—	—
XML W3C Recommendation, XML Digital Signature	—	—

* Соответствующий национальный стандарт отсутствует. До его утверждения рекомендуется использовать перевод на русский язык данного международного стандарта (документа).

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

- IDT — идентичные стандарты.

Библиография

[1] ISO 3166-1, Codes for the representation of names of countries and their subdivisions — Part 1: Country codes
[2] ISO/IEC 10646, Information technology — Universal Multiple-Octet Coded Character Set (UCS)
[3] ISO/IEC 11179 (all parts), Information technology — Metadata registries (MDR)
[4] ISO 13606 (all parts), Health informatics — Electronic health record communication
[5] CEN/TS 14796, Health informatics — Data types
[6] IETF RFC 2978, IANA Charset Registration Procedures
[7] OASIS CIQ, http://www.oasis-open.org/committees/ciq
[8] International Health Terminology Standards Development Organisation, SNOMED Clinical Terms, Abstract Logical Model and Representational Forms, November 2006. Revision to External Draft Guide [viewed 27 August 2010] Available from: http://gost.e-ecolog.ru representational_forms.pdf
[9] ICD-9 International Statistical Classification of Diseases and Related Health Problems, Ninth Revision, Instruction Manual. Available from: gost.e-ecolog.ru
[10] ICD-10 International Statistical Classification of Diseases and Related Health Problems, Tenth Revision, Instruction Manual, 2nd ed. [viewed 27 August 2010]. Available from: gost.e-ecolog.ru ed_volume2.pdf

УДК 004:61.006.354 ОКС 35.240.80 П85 ОКСТУ 4002

Ключевые слова: здравоохранение, информатизация здоровья, электронная передача данных, тип данных

Редактор *А.Ф. Колчин*
Корректор *Е.Р. Ароян*
Компьютерная верстка *Ю.В. Половой*

Сдано в набор 08.12.2016. Подписано в печать 20.01.2017. Формат 60 × 84 1/8. Гарнитура Ариал.
Усл. печ. л. 21,86.

Подготовлено на основе электронной версии, предоставленной разработчиком стандарта

Набрано в ИД «Юриспруденция», 115419, Москва, ул. Орджоникидзе, 11.
www.jurisizdat.ru gost.e-ecolog.ru

Издано во ФГУП «СТАНДАРТИНФОРМ», 123995, Москва, Гранатный пер., 4
gost.e-ecolog.ru gost.e-ecolog.ru

gost.e-ecolog.ru