ФЕДЕРАЛЬНОЕ АГЕНТСТВО

ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р ИСО 13584-32— 2012

Системы промышленной автоматизации и интеграция

БИБЛИОТЕКА ДЕТАЛЕЙ

Часть 32

Ресурсы практической реализации. Язык онтологической разметки продукции

ISO 13584-32:2010
Industrial automation systems and integration — Parts library — Part 32: Implementation resources: OntoML: Product ontology markup language (IDT)

Издание официальное

Москва
Стандартинформ
2015

Предисловие

1 ПОДГОТОВЛЕН АНО «Международная академия менеджмента и качества бизнеса» на основе собственного аутентичного перевода на русский язык международного стандарта, указанного в пункте 4

2 ВНЕСЕН Техническим комитетом по стандартизации ТК 100 «Стратегический и инновационный менеджмент»

3 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Приказом Федерального агентства по техническому регулированию и метрологии от 29 ноября 2012 г. № 1702-ст

4 Настоящий стандарт идентичен международному стандарту ИСО 13584-32:2010 «Системы промышленной автоматизации и интеграция. Библиотека деталей. Часть 32. Ресурсы практической реализации. Язык онтологической разметки продукции» (ISO 13584-32:2010 «Industrial automation systems and integration — Parts library — Part 32: Implementation resources: OntoML: Product ontology markup language»).

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

5 ВВЕДЕН ВПЕРВЫЕ

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

Содержание

1 Область применения . . . 1
2 Нормативные ссылки . . . 1
3 Термины и определения . . . 2
4 Аббревиатуры . . . 6
5 Уровни практической реализации OntoML-языка . . . 6
6 Обзор представлений OntoML-онтологии . . . 6
6.1 Понятия CIIM-онтологии . . . 6
6.2 Структура OntoML-языка для понятий CIIM-онтологии . . . 7
6.3 Графическое UML-представление конструкций OntoML-языка . . . 7
6.4 Общая структура OntoML-языка . . . 15
6.5 Заголовок в OntoML-языке . . . 16
6.6 Корневой элемент онтологии . . . 18
6.7 OntoML-представление понятий CIIM-онтологии . . . 19
7 Общие сведения о представлении OntoML-библиотек . . . 47
7.1 Корневой элемент библиотеки . . . 48
7.2 Общая структура расширений класса . . . 48
7.3 Библиотека простого уровня: содержание классов продукции . . . 51
7.4 Библиотека повышенного уровня: содержание классов представления продукции . . . 53
8 Прочие структурированные информационные элементы . . . 55
8.1 Переводы . . . 55
8.2 Внешний контент . . . 58
8.3 Система типов данных . . . 64
8.4 Единицы измерений . . . 86
8.5 Ограничительные условия . . . 91
8.6 Алостериорное семантическое соотношение . . . 104
8.7 Идентификация спецификации на обмен данными . . . 108
8.8 Другие структурированные информационные элементы . . . 109
9 Структура обмена OntoML-данными . . . 111
9.1 Идентификаторы онтологических CIIM-понятий . . . 111
9.2 Область OntoML-имен . . . 116
9.3 Модульная структура OntoML-языка . . . 116
9.4 Уровни обмена данными и классы соответствия . . . 117
9.5 Требование к классам соответствия . . . 118
10 Правила управления внесением изменений в словарь . . . 124
10.1 Принцип онтологической непрерывности . . . 124
10.2 Редакции и версии словаря . . . 125
10.3 Исправление ошибок . . . 127
10.4 Правила управления внесением изменений . . . 128
10.5 Изменения и атрибуты словаря . . . 132
10.6 Ограничительные условия на изменение словарей-справочников . . . 132
Приложение А (обязательное) Регистрация информационных объектов . . . 134
Приложение В (справочное) Компьютерно-интерпретируемые списки . . . 135
Приложение С (справочное) Требования к стандартным данным в OntoML-языке . . . 136
Приложение D (справочное) Представление значений величин в ИСО 13584/МЭК 61360 и типов данных в совместно используемых в ИСО/ТС 29002-10 схемах . . . 137
Приложение Е (справочное) Онтологическое описание расширенных значений в OntoML-языке . . . 160
Приложение F (справочное) Преобразование структуры CIIM-модели из XML-структуры OntoML-языка на язык EXPRESS . . . 166
Приложение G (справочное) Уровни обмена экземплярами OntoML-документа . . . 200
Приложение Н (справочное) Перечень форматов значений . . . 202
Приложение I (справочное) Пример XML-файла . . . 214
Приложение J (справочное) Информация для поддержки внедрения настоящего стандарта . . . 221
Приложение ДА (справочное) Сведения о соответствии ссылочных международных стандартов ссылочным национальным стандартам Российской Федерации . . . 222
Библиография . . . 223

Введение

Комплекс международных стандартов ИСО 13584 разработан с целью компьютерного представления данных, содержащихся в библиотеке деталей, а также для обмена этими данными. Целью настоящего стандарта является представление объективного способа, обеспечивающего передачу данных о библиотеках деталей и не зависящего от любого назначения системы, в которой эти данные будут использоваться. Благодаря характеру описания оно может быть использовано не только для обмена файлами, содержащими детали, но и в качестве основы для реализации и совместного использования баз данных для элементов библиотеки деталей.

Комплекс международных стандартов ИСО 13854 выполнен в виде частей, каждая из которых публикуется отдельно. Части комплекса международных стандартов ИСО 13854 попадают в одну из следующих категорий: концептуальные описания, ресурсы программной реализации, методология описаний, проверка на соответствие, протокол просмотра и обмена данными и стандартизованное содержание. Данные категории описаны в ИСО 13584-1. Настоящий стандарт является одной из частей, посвященных методологии описаний.

Настоящий стандарт содержит правила и руководства для технических комитетов по стандартизации и поставщиков информации с целью создания онтологий продукции. Онтологии продукции состоят из иерархии характеристических классов деталей, выполненных в соответствии с общей методологией и обеспечивающих согласованность между поставщиками. Правила и руководства, приведенные в настоящем стандарте, содержат методы группировки деталей в характеристические классы деталей с целью образования иерархий; методы сопоставления свойств деталей с характеристическими классами деталей и со словарем элементов, содержащим классы и свойства деталей.

Настоящий стандарт можно рассматривать в качестве нормативной ссылки на модель данных, определяющую обмен данными словаря. EXPRESS-спецификация разработана как общая модель для ИСО 13584 и МЭК 61360 и опубликована в качестве стандарта МЭК 61360-2. Настоящий стандарт применяет рассматриваемые ниже концепции к общей модели.

Комплекс международных стандартов ИСО 13584 имеет общее название «Системы промышленной автоматизации и интеграция. Библиотека деталей» и включает в себя следующие части:

- часть 1. Общие положения и фундаментальные принципы;
- часть 10. Концептуальное описание. Концептуальная модель библиотеки компонентов;
- часть 20. Логические ресурсы. Логическая модель выражений;
- часть 24. Логические ресурсы. Логическая модель библиотеки поставщика;
- часть 26. Логические ресурсы. Идентификация поставщика;
- часть 31. Ресурсы программной реализации. Интерфейс геометрического программирования;
- часть 42. Методология описания. Методология структурирования семейств компонентов;
- часть 101. Протокол обмена и просмотра данных. Геометрический протокол обмена и просмотра данных с помощью параметрической программы;
- часть 102. Протокол обмена и просмотра данных. Протокол обмена и просмотра данных на основе спецификации соответствия ИСО 10303.

Структура комплекса международных стандартов приведена в ИСО 13584-1. Нумерация частей комплекса отражает его структуру:

- части 10—19 устанавливают концептуальные описания;
- части 20—29 устанавливают логические ресурсы;
- части 30—39 устанавливают ресурсы программной реализации;
- части 40—49 устанавливают методологию описания;
- части 50—59 устанавливают требования к проведению проверки на соответствие;
- части 100—199 устанавливают требования к оформлению протоколов просмотра и обмена данными;
- части 500—599 устанавливают стандартизованное содержание.

Полный перечень стандартов комплекса ИСО 13584 приведен в сети Интернет.

М Е Ж Г О С У Д А Р С Т В Е Н Н Ы Й С Т А Н Д А Р Т

Системы промышленной автоматизации и интеграция

БИБЛИОТЕКА ДЕТАЛЕЙ

Ч а с т ь 32

Ресурсы практической реализации. Язык онтологической разметки продукции

Industrial automation systems and integration.
Parts library. Part 32. Implementation resources.
Product ontology markup language

Дата введения — 2014—01—01

1 Область применения

В настоящем стандарте приведено описание XML-языка и XML-схемы для представления данных в соответствии с моделью данных согласно настоящему стандарту.

Настоящий стандарт распространяется на:

- представление общей модели ИСО 13584/МЭК 61360 с помощью языка UML;
- определение двух уровней практической реализации общей модели ИСО 13584/МЭК 61360, названных простым и сложным уровнем соответственно;
- спецификацию положений XML-языка, обеспечивающую обмен в XML-формате простых и сложных онтологий, соответствующих общей модели ИСО 13584/МЭК 61360;
- спецификацию положений XML-языка, обеспечивающую обмен простых и сложных онтологий, соответствующих общей модели ИСО 13584/МЭК 61360, а также семейств продукции, чьи характеристики определяются с помощью указанных онтологий.

П р и м е ч а н и е 1 — В настоящем стандарте такой контекст обмена назван библиотекой OntoML.

П р и м е ч а н и е 2 — Информационные модели обмена семейств продукции, характеристики которых устанавливаются с помощью онтологий, соответствующих общей модели ИСО 13584/МЭК 61360, определены в ИСО 13584-25;

- спецификацию глобальных элементов XML-языка, позволяющую использовать OntoML-язык как формат обмена для представления ответов на запросы, выполненных с помощью механизма идентификации концептуального словаря ИСО 29002-20;
- спецификацию формального отображения, позволяющую ассоциировать все элементы языка OntoML, атрибуты соответствующих объектов и атрибуты общей модели ИСО 13584/МЭК 61360 с моделью данных на EXPRESS-языке.

Настоящий стандарт не распространяется на:

- правила построения положений OntoML-языка с помощью общей модели ИСО 13584/МЭК 61360;
- спецификацию программы, предназначенную для интерпретации всех операторов отображения, определенных в OntoML-языке для построения соответствующих представлений общей модели ИСО 13584/МЭК 61360 на EXPRESS-языке в соответствии с ИСО 10303-21;
- обмен индивидуальных продуктов, характеристики которых определены с помощью онтологий, соответствующих общей модели ИСО 13584/МЭК 61360.

П р и м е ч а н и е 3 — Для обмена данными об индивидуальной продукции можно использовать формат обмена данными, определенный в ИСО/ТС 29002-10.

2 Нормативные ссылки

В настоящем стандарте использованы нормативные ссылки на следующие стандарты, которые необходимо учитывать при использовании настоящего стандарта. В случае ссылок на документы, у которых указана дата утверждения, необходимо пользоваться только указанной редакцией. В случае, когда дата

Издание официальное

утверждения не приведена, следует пользоваться последней редакцией ссылочных документов, включая любые поправки и изменения к ним:

ИСО 10303-11:2004 Системы промышленной автоматизации и интеграция. Представление данных о продукции и обмен данными. Часть 11. Методы описания. Справочное руководство по языку EXPRESS (ISO 10303-11:2004, Industrial automation systems and integration — Product data representation and exchange — Part 11: Description methods: The EXPRESS language reference manual)

ИСО/МЭК 14977:1996 Информационные технологии. Синтаксический метаязык. Расширенная БНФ (ISO/IEC 14977:1996, Information technology — Syntactic metalanguage — Extended BNF)

ИСО/ТС 29002-5:2009 Промышленные автоматические системы и интеграция. Обмен характеристическими данными. Часть 5. Схема идентификации (ISO/TS 29002-5:2009 Industrial automation systems and integration — Exchange of characteristic data — Part 5: Identification scheme)

ИСО/ТС 29002-10:2009 Промышленные автоматические системы и интеграция. Обмен характеристическими данными. Часть 10. Формат обмена характеристическими данными (ISO/TS 29002-10:2009, Industrial automation systems and integration — Exchange of characteristic data — Part 10: Characteristic data exchange format)

Многоцелевые расширения электронной почты Интернет. Часть 1: Формат сообщений в Интернете. Инженерная целевая рабочая группа Интернета RFC 2045. Ноябрь 1996 г. [цитировано 15 августа 2000]. Доступно на сайте: http://www.ietf.org/rfc/rfc2045.txt

Единые идентификаторы ресурсов (URI): Обобщенный синтаксис. Инженерная целевая рабочая группа RFC 2396. Август 1998 г. [цитировано 7 августа 2000 г.]. Доступно на сайте: http://www.ietf.org/rfc/rfc2396.txt

Расширенный язык разметки (XML) 1.0. Четвертое издание. Рекомендации в редакции Интернет-Консорциума от 14 июня 2006 г. Доступны на сайте: <http://www.w3.org/TR/2006/PER-xml-20060614>

XML-схема данных. Часть 1: Структуры. Второе издание. Рекомендации Интернет-Консорциума от 28 октября 2004 г. Доступны на сайте: <http://www.w3.org/TR/2004/REC-xmlschema-1-20041028>

XML-схема данных. Часть 2: Типы данных. Второе издание. Рекомендации Интернет-Консорциума от 28 октября 2004 г. Доступны сайте: <http://www.w3.org/TR/2004/REC-xmlschema-1-20041028>

XML-язык маршрутизации (XPath) 1.0. Рекомендации Интернет-Консорциума от 16 ноября 1999 г. Доступны на сайте: <http://www.w3.org/TR/1999/REC-хмаршрут-19991116>

Пространства имен в XML-языке 1.0. Второе издание. Рекомендации Интернет-Консорциума от 14 июня 2006 г. Доступны на сайте: <http://www.w3.org/TR/2006/PER-xml-им-20060614>

3 Термины и определения

В настоящем стандарте применены следующие термины с соответствующими определениями:

3.1 **класс** (class): Абстрактное понятие для множества аналогичных продуктов.

Примечание — Заимствовано из ИСО 13584-42:2010, определение 3.6.

3.2 **член (элемент) класса** (class member): Объект, соответствующий абстрактному понятию, определенному в некотором классе.

[ИСО 13584-42:2010, определение 3.8]

3.3 **общая модель ИСО 13584/МЭК 61360** (common ISO 13584/IEC 61360 model): Модель данных для описания онтологии продукции, построенная с помощью EXPRESS-языка моделирования информации в соответствии со стандартами ИСО/ТК 184/ПК4/РГ2 и МЭК/ПК3D.

Примечание 1 — заимствовано из ИСО 13584-42:2010, определение 3.10.

Примечание 2 — Предшествующая версия общей модели онтологии ИСО 13584/МЭК 61360 опубликована в МЭК 61360-5 и ИСО 13584-25:2004. Новая версия, соответствующая данной версии OntoML и ИСО 13584-42:2010, разрабатывается в настоящее время.

3.4 **понятие CIIM-онтологии** (CIIM ontology concept): Базовая единица знаний, представленная в онтологии, основанной на общей модели онтологии ИСО 13584/МЭК 61360.

Примечание 1 — Понятия CIIM-онтологии — это источники (поставщики) информации, классы, свойства, типы данных и документы.

Примечание 2 — Каждое понятие CIIM-онтологии ассоциируется с глобальным идентификатором, обеспечивающим внешнюю ссылку на файл обмена.

Примечание 3 — На одно и то же понятие CIIM-онтологии можно ссылаться несколько раз в одном и том же файле обмена. Поэтому ссылочный механизм определен в OntoML-языке.

3.5 **атрибут EXPRESS-языка** (EXPRESS attribute): Элемент данных для компьютерного описания свойства, соотношения или класса.

Примечание — Атрибут описывает только отдельные характеристики свойства, класса или соотношения.

Пример — Имя свойства, код класса, единицы измерения, в которых представляются значения свойств — примеры атрибутов.

3.6 **сущность EXPRESS-языка** (EXPRESS entity): Класс информации, определенный общими свойствами.

[ИСО 10303-11:1994, определение 3.2.5]

3.7 **глобальный идентификатор** (global identifier): Код, обеспечивающий однозначную и универсальную уникальную идентификацию некоторых понятий или объектов.

Примечание — Все понятия CIIM-онтологии, ассоциированные с глобальным идентификатором.

3.8 **экземплярное соотношение** (is-a relationship): Соотношение включения в класс, связанное с наследованием.

Примечание 1 — Если класс А1 представляет класс А, то каждая продукция, принадлежащая классу А1, принадлежит классу А, и все описанное в контексте класса А будет автоматически дублироваться в контексте класса А1.

Примечание 2 — Данный метод обычно называется "наследованием".

Примечание 3 — В общей словарной модели ИСО 13584/МЭК 61360 экземплярные соотношения могут определяться только между классами характеристик. Рекомендуется, чтобы они определяли отдельные иерархии и гарантировали, что наследуются явные и применимые свойства.

[ИСО 13584-42:2010, определение 3.23]

3.9 **условное соотношение** (is-case-of relationship): Механизм импортирования свойств.

Примечание 1 — Если класс А1 является условным для класса А, то определение продукции класса А также распространяется и на продукцию класса А1 (таким образом, класс А1 может импортировать любое свойство из класса А).

Примечание 2 — Цель условного соотношения — разрешить соединение нескольких иерархий включений множеств при условии, что ссылочные иерархии могут обновляться независимо.

Примечание 3 — Не существует ограничений, определяющих, что условное соотношение предназначено для определения отдельной иерархии.

Примечание 4 — В общей модели словаря ИСО 13584/МЭК 61360 условные соотношения могут быть использованы в следующих четырех случаях: (1) для связи класса характеристик с классом категорий, (2) для импортирования (в контексте некоторых стандартизованных ссылочных словарей) некоторых свойств, уже определенных в других стандартизованных ссылочных словарях, (3) для соединения ссылочного словаря пользователя с одним или несколькими другими стандартизованными ссылочными словарями, (4) для описания продуктов одного класса с помощью свойств другого класса: если продукты класса А1 выполняют две различные функции, и, таким образом, логически описываются свойствами, ассоциированными с двумя различными классами А и В, то класс А1 может быть, например, присоединен экземплярным соотношением к классу А и условным соотношением — к классу В.

Примечание 5 — Заимствовано из ИСО 13584-42:2010, определение 3.24.

3.10 **производное соотношение** (is-view-of): Соотношение, обеспечивающее формальное выражение того факта, что один объект является представлением другого объекта в соответствии с заданной перспективой.

Пример — Набор геометрических объектов может давать приближенное представление некоторого винта. Если набор геометрических объектов и винт представлены как объект, то рассматриваемое производное соотношение находится между первым объектом и последним объектом (в эскизной перспективе).

[ИСО 13584-24:2003, определение 3.64]

3.11 **библиотека** (library): Представление множества различной продукции характеристиками, возможно ассоциированными с онтологией, где определены классы характеристик продукции и ее свойства.

Примечание 1 — Библиотеки также называют каталогами.

Примечание 2 — В схемах OntoML-языка необходимо различать элементы онтологии и элементы содержания. Элементы онтологии встроены в XML-элементы "словаря", а элементы содержания — в XML-элементы "библиотеки".

3.12 **реализация OntoML-документа** (OntoML document instance): XML-документ, удовлетворяющий требованиям XML- схемы для OntoML-языка.

3.13 **категоризация продукта, категоризация детали, категоризация** (product categorization, part categorization, categorization): Рекурсивное разделение множества различной продукции на подмножества для достижения особых целей.

Примечание 1 — Подмножества, используемые в категоризации продукта, называются классами категоризации продукции или категориями продукции.

Примечание 2 — Категоризация продукции не является ее онтологией. Она не может быть использована для определения характеристик продукции.

Примечание 3 — Никакие свойства не связываются с категориями.

Примечание 4 — Возможны несколько категоризаций одного и того же множества продукции в соответствии с их целевым использованием.

Пример — Классификация UNSPSC США является примером категоризации продукции (используется при анализе расходов).

Примечание 5 — С помощью условного соотношения несколько иерархий класса характеристик продукции могут быть присоединены к иерархии категорий для получения отдельной структуры.

[ИСО 13584-42:2010, определение 3.32]

3.14 **класс категоризации продукции, класс категоризации деталей, класс категоризации** (product categorization class, part categorization class, categorization class): Класс продукции, представляющий собой элемент категоризации.

Пример — Производственные компоненты, поставки и промышленная оптика — примеры класса категоризации продукции, определенного в UNSPSC-классификаторе.

Примечание 1 — Настоящий стандарт не определяет правила отбора классов категоризации. Данное понятие введено, чтобы: (1) уяснить его отличие от класса характеристик, и (2) пояснить, что один и тот же класс характеристик может быть объединен с любым количеством классов категоризации.

Примечание 2 — Не существует свойств, ассоциированных с классом категоризации.

[ИСО 13584-42:2010, определение 3.33]

3.15 **характеризация продукции, характеризация детали** (product characterization, part characterization): Описание продукта с помощью класса характеристик продукции, которому он принадлежит, и множества пар значений свойств.

Пример — Свойство Hexagon_head_bolts_ISO_4014 (класс точности продукта = A, тип резьбы = M, длина = 50, диаметр = 8) — пример характеристики продукта.

Примечание — В примере выше свойство Hexagon_head_bolts_ISO_4014 является идентификатором класса характеризации продукции "Болты с шестигранной головкой", определенного в ИСО 4014. Все имена, выделенные курсивом в скобках, являются идентификаторами свойств болтов, определенных в ИСО 4014.

[ИСО 13584-42:2010, определение 3.34]

3.16 **класс характеризации продукции, класс характеризации деталей, класс характеризации** (product characterization class, part characterization class, characterization class): Класс продукции, выполняющий ту же функцию и разделяющий общие свойства.

Примечание — Класс характеризации продукции может быть определен с разной степенью детализации, определяя, таким образом, иерархию включения множеств.

Пример — Болт/винт с метрической резьбой и болт с шестигранной головкой — примеры классов характеризации продуктов, определенных в ИСО 13584-511. Первый класс характеристик включается во второй. Транзистор и двухполюсный силовой транзистор — примеры классов характеризации продуктов, определенных в МЭК 61360-4-DB. Второй класс включается в первый.

[ИСО 13584-42:2010, определение 3.35]

3.17 **онтология продукции, онтология детали, онтология** (product ontology, part ontology, ontology): Модель знаний о продукции, полученная путем формального и согласованного представления понятий области (домена) продукции в терминах идентифицированных классов характеризации, соотношений классов и идентифицированных свойств.

Примечание 1 — Онтологии продукции основаны на модели реализации класса, которая позволяет распознать и обозначить виды продукции, называемые классами характеризации, имеющие аналогичные функции (например, *шариковые подшипники, конденсаторы*), а также провести различие внутри класса между подмножествами продуктов, называемыми реализациями и считающимися идентичными. Для формулировки обозначений и определений классов характеризации рекомендуется использовать правила, определенные в ИСО 1087-1. Реализации не имеют определений. Они обозначаются классом, которому они принадлежат и множеством пар значений свойств.

Примечание 2 — Онтологии связаны не со словами, а с понятиями. Они не зависят от выбора языка.

Примечание 3 — "Согласованность понятий" означает, что данные понятия согласованы в некотором сообществе пользователей.

Примечание 4 — Термин "формальный" означает, что онтология предназначена для компьютерной интерпретации. Существует уровень компьютерного мышления, способный воспринять онтологию (например, выполнить проверку согласованности, сделать вывод).

Пример 1 — Проверка согласованности — вид компьютерного мышления.

Примечание 5 — "Идентифицированный" означает, что все классы характеризации онтологии и свойства ассоциированы с глобально уникальным идентификатором, позволяющим ссылаться на данное понятие из любого контекста.

Примечание 6 — В OntoML-языке онтологии называются сложными, если они используют все механизмы моделирования, определенные в общей модели онтологии ИСО 13584/МЭК 61360. OntoML-язык также определяет простое функциональное подмножество данной модели, позволяющее определять простые онтологии.

Примечание 7 — В настоящем стандарте каждая онтология продукта обращается к заданной области продукции, соответствующей общей модели словаря ИСО 13584/МЭК 61360, называемой ссылочным словарем для указанной области.

Пример 2 — Понятие онтологии продукции определено в МЭК 61360. Оно согласовано всеми официальными членами группы МЭК/ПК3D. Корпоративная онтология согласуется экспертами, назначаемыми руководством компании от ее имени.

Примечание 8 — Заимствовано из ИСО 13584-42:2010, определение 3.36.

3.18 **свойство** (property): Определяет параметр, необходимый для описания и дифференциации продукции.

Примечание 1 — Свойство описывает один аспект данного объекта.

Примечание 2 — Свойство определяется тотальностью его ассоциированных атрибутов. Типы и количество атрибутов, описывающих свойство, с высокой точностью описаны в настоящем стандарте.

Примечание 3 — Термин "свойство", использованный в настоящем стандарте, и термин "тип элемента данных", использованный в МЭК 61360, — синонимы.

Примечание 4 — Заимствовано из ИСО 13584-42:2010, определение 3.37.

3.19 **ссылочный словарь** (reference dictionary): Онтология продукции, соответствующая общей модели словаря ИСО 13584/МЭК 61360.

Примечание — В серии стандартов ИСО 13584 онтология продукции, относящаяся к заданной области продукции, основана на общей модели словаря ИСО 13584/МЭК 61360, называемой ссылочным словарем для указанной области.

[ИСО 13584-42:2010, определение 3.41]

3.20 **атрибут XML-языка** (XML attribute): Конструкция XML-языка, включенная в элемент и определенная именем и простой парой значений.

Примечание 1 — Заимствовано из рекомендаций к версии XML 1.0.

Примечание 2 — В настоящем стандарте имя атрибута XML-языка имеет приставку "@". Это означает, что соответствующий блок информации представлен как атрибут, а не как встроенный XML-элемент.

3.21 **комплексный тип XML-языка** (XML complex type): Множество элементов XML-языка и/или положений атрибутов, описывающих модели, содержащие элементы XML-языка.

3.22 **элемент XML-языка** (XML element): Структура XML-языка, включающая метку начала, метку окончания, информацию между указанными метками и, возможно, множество атрибутов XML-языка.

Примечание 1 — Заимствовано из рекомендаций к версии XML 1.0.

Примечание 2 — Информационная структура, расположенная между метками (тегами) начала и окончания, имеет либо простой XML-тип данных, либо комплексный XML-тип.

Примечание 3 — Один элемент XML-языка может содержать другие элементы XML-языка, имеющие либо простой, либо комплексный XML-тип данных.

3.23 **простой XML-тип данных** (XML simple type): Множество ограничений, применимых к значению атрибута XML-языка или к значению элементов XML-языка без каких-либо дочерних XML-элементов.

Примечание — Простой XML-тип данных применяется к значениям атрибутов и только к текстовому содержанию элементов.

4 Аббревиатуры

CIIM — Общая модель ИСО 13584/МЭК 61360 (Common ISO 13584/IEC 61360 Model);
IRDI — Международный идентификатор регистрации данных (International Registration Data Identifier);
SI — Международная система единиц СИ (Système International d'Unités (International System of Units));
STEP — Стандарт обмена модельными данными продукции (Standard for The Exchange of Product model data);
UNSPSC — Классификация продукции и услуг, определенная ООН (classification of products and services defined by the United Nations);
URI — Унифицированный идентификатор ресурса (Uniform Resource Identifier);
URN — Унифицированное имя ресурса (Uniform Resource Name);
UML — Унифицированный язык моделирования (Unified Modeling Language);
XML — Расширенный язык разметки (Extensible Markup Language).

5 Уровни практической реализации OntoML-языка

CIIM-модель включает ряд понятий и методов моделирования, позволяющих характеризовать не только элементы (продукты), но также: (1) представления элементов с нескольких точек зрения, (2) характеристики этих различных возможных точек зрения. Данные сложные понятия могут не использоваться в ряде приложений.

Настоящий стандарт идентифицирует подмножество всех механизмов моделирования, определенных моделью CIIM. Это может оказаться полезным для контекстов ряда приложений. Данное подмножество определяет допустимые уровни практической реализации OntoML-языка. Указанные уровни определены как "простые" в настоящем стандарте.

Все механизмы моделирования, не принадлежащие указанному «простому» уровню (если они имеются), рассматриваются как "сложные". В 9.5 приведены указанные OntoML-конструкции, принадлежащие как к простому, так и к сложному уровню.

Примечание 1 — Простые уровни определяются как устойчивые функциональные подмножества. Для понимания и использования простых уровней нет необходимости использовать сложные уровни чтения и восприятия.

OntoML-язык дает возможность моделировать два вида информации:

- онтологии;
- библиотеки, являющиеся множеством реализации данных, возможно ассоциированных с определением их онтологии.

Не все указанные виды информации (в зависимости от контекста приложения) могут оказаться полезными. Поэтому четыре OntoML-подмножества определены как допустимые уровни практической реализации:

- простая онтология;
- сложная онтология;
- простая библиотека;
- сложная библиотека.

Простой уровень реализации данных не существует. Данный вид обмена информацией определяется ИСО/ТС 29002-10.

Примечание 2 — ИСО/ТС 29002 — это совместная разработка нескольких комитетов по стандартизации. Она способствует взаимодействию различных стандартов, определяющих характеристики продуктов.

Примечание 3 — Только простая онтология и подмножества сложной онтологии соответствуют для модели CIIM. Представление библиотек соответствует расширениям CIIM-модели, определенным в ИСО 13584-24:2003 и ИСО 13584-25:2004.

6 Обзор представлений OntoML-онтологии

В данном разделе определены понятия CIIM-онтологии и представлена их базовая структура. Для иллюстрации каждой структуры понятия CIIM-онтологии использованы графические обозначения.

6.1 Понятия CIIM-онтологии

В соответствии с CIIM-моделью онтология включает пять видов основных понятий:

- поставщик;

- класс;
- свойство;
- идентифицированные типы данных;
- документ.

Каждое указанное понятие CIIM-онтологии включает два вида информации:

- глобальный идентификатор. Данный идентификатор позволяет ссылаться на данное понятие изнутри или извне OntoML-документа, что и определяет рассматриваемое понятие. Структура глобальных идентификаторов понятий CIIM-онтологии определена в 9.1.

Примечание — Глобальный OntoML-идентификатор для понятия CIIM-онтологии содержит ту же информацию, что и идентификаторы, определенные в других частях комплекса международных стандартов ИСО 13584 (далее — ИСО 13584). Он является базовой семантической единицей;

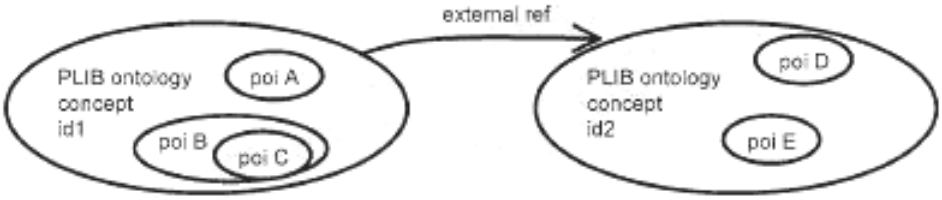
- его определение, включающее несколько блоков информации, описанных в CIIM-модели.

6.2 Структура OntoML-языка для понятий CIIM-онтологии

Каждое определение понятия CIIM-онтологии включает несколько блоков информации и соотношений, содержащих другие понятия онтологии CIIM. В представлении XML-языка каждое понятие CIIM-онтологии представлено одним XML-элементом. При этом блоки информации, способствующие определению понятия CIIM-онтологии, представляются либо внешне, либо внутренне для соответствующего ассоциированного элемента XML-языка в зависимости от использованного соотношения:

- внешние представления используются для ссылок на любой блок информации, представляющий другое понятие CIIM-онтологии, через его собственный идентификатор;
- внутренние представления используются для ссылок на любой другой блок информации.

Рисунок 1 иллюстрирует указанные два вида представления.



poi: piece of information

external ref	Внешние ссылки
PLIB ontology concept id1	Понятие онтологии библиотеки PLIB. Идентификатор id1
poi A	Блок информации A
poi: piece of information	poi — блок информации

Рисунок 1 — Описание понятий CIIM-онтологии

Допустим, что определены два понятия CIIM-онтологии. Они оба однозначно идентифицированы (их идентификаторы: *id1*, *id2*). Дополнительно они оба определены несколькими блоками информации (*poi*), встроенными внутрь XML-представления понятий CIIM-онтологии. С другой стороны, указанные блоки информации могут сами включать некоторые другие блоки информации. Наконец, понятие CIIM-онтологии с идентификатором *id1* ссылается на понятие CIIM-онтологии с идентификатором *id2*.

Примечание — Использование внутреннего представления (для внедрения ссылочного блока информации внутрь элемента XML-языка, представляющего понятие онтологии модели CIIM) может привести к дублированию некоторых блоков информации. В любом случае это не изменяет семантики базовой CIIM-модели данных на EXPRESS-языке.

6.3 Графическое UML-представление конструкций OntoML-языка

В настоящем стандарте OntoML-язык описывается с помощью обозначений языка UML. Базовые UML-обозначения расширяются для:

- выявления различий между элементами XML-языка и атрибутами XML-языка;

- явного представления ссылок на комплексный XML-тип данных, представленных в различных диаграммах;
- представления ссылок на понятия CIIM-онтологии с помощью описанного метода идентификации;
- упрощения диаграмм при представлении ссылок на понятия CIIM-онтологии.

Указанные графические обозначения называются UML-обозначениями.

Данный раздел содержит графические UML-представления конструкций OntoML-языка. Он также описывает механизм, используемый для представлений на OntoML-языке ссылок на понятия CIIM-онтологии вместе с их графическими представлениями.

После представления ссылочного механизма, используемого для организации внешних ссылок на понятия CIIM-онтологии, данный раздел представляет структуру различных понятий CIIM-онтологии, определенных в OntoML-языке, с помощью UML-диаграмм.

6.3.1 Графические обозначения

Далее в настоящем стандарте для представления OntoML-структуры с помощью UML-диаграмм будут использованы нижеследующие соглашения.

6.3.1.1 Представление комплексного XML-типа данных

Комплексный XML-тип данных на схеме представляется как прямоугольник, разделенный на две части: сверху — комплексный тип XML, снизу — атрибут XML-языка и/или встроенный элемент XML-языка (см. рисунок 2).

Пример 1 — На рисунке 2 представлен комплексный XML-тип данных. Он называется PROPERTY_Type (тип свойства).

Рисунок 2 — UML-представление комплексного XML-типа данных

Если рассматриваемый комплексный XML-тип данных является абстрактным, то его имя приводится *курсивом*.

Комплексный тип может также быть представлен на схеме прямоугольником со скругленными углами. Это означает, что атрибуты XML-языка и/или встроенные элементы XML-языка, описывающие его модель содержания, ранее уже где-то были определены.

Пример 2 — Рисунок 3 описывает ссылку на комплексный XML-тип данных PROPERTY_Type.

Рисунок 3 — UML-представление ссылки на комплексный XML-тип данных

6.3.1.2 Представление ссылок на внешние информационные элементы

OntoML-язык использует ресурсы внешней XML-схемы для определения его собственного содержания. Для этой цели вводятся графические обозначения. Рассматривается ссылка на комплексный XML-тип данных. Поэтому, во-первых, на рисунке она представлена прямоугольником со скругленными углами. Во-вторых, это внешняя ссылка. По этой же причине данный прямоугольник закрашен светло-серым цветом (см. рисунок 4 ниже).

Пример — На рисунке 4 на некоторый комплексный тип Content (Содержание) производится ссылка из другой XML-схемы, идентифицированной приставкой cat.

Рисунок 4 — UML-представление внешней ссылки на комплексный XML-тип данных

Примечание — Приставка определяется методом формирования области XML-имен. Она позволяет распознавать определения XML, описанные в других словарях внешней XML-схемы.

6.3.1.3 Представление атрибутов и элементов XML-языка, модель контента которых является простым XML-типом данных.

Атрибуты и элементы XML-языка, содержание которых является простым типом XML, встроенным внутрь комплексного типа XML, представляются их именем и типом. Для распознавания атрибутов XML-языка и (вложенных) XML-элементов их имена выделяются приставкой "@".

Примечание — Символ '@' не является частью имени атрибута. Следовательно, данный символ не представлен в XML-схеме на OntoML-языке.

На рисунке 5 ниже тип **PROPERTY_Type** является абстрактным комплексным XML-типом данных. Он содержит встроенный элемент **revision**, типом которого является простой XML-тип **REVISION_TYPE_type**. Указан также атрибут языка XML **«id»**, типом которого является простой XML-тип **PropertyId**.

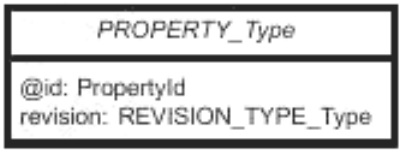


Рисунок 5 — UML-представление атрибутов XML-языка и элементов простого типа XML-языка

Пример — На рисунке 6 приведен текст XML-программы, соответствующий рисунку 5.

```
<xs:complexType name="PROPERTY_Type" abstract="true">
    <xs:sequence>
        <xs:element name="revision" type="REVISION_TYPE_Type"/>
    </xs:sequence>
    <xs:attribute name="id" type="PropertyId" use="required"/>
</xs:complexType>
```

Рисунок 6 — Представление XML для атрибутов языка XML и элементов языка XML простого типа

6.3.1.4 Представление элементов XML-языка, модель контента которых является комплексным XML-типом данных.

Элементы XML-языка, модели контента (содержания) которых являются комплексными XML-типами данных, представлены как соотношение между элементом XML-языка комплексного типа и комплексным типом, включающим модель содержания данного элемента XML-языка. Рассматриваемое соотношение представлено закрашенным ромбом и прямой со стрелкой. Данный ярлык, ассоциированный с рассматриваемым соотношением, представляет имя элемента XML-языка.

Примечание 1 — Закрашенный ромб указывает на составное соотношение.

Примечание 2 — По умолчанию кардинальное число прямого соотношения равно одному.

Пример — На рисунке 7 определен элемент XML-языка domain. Это встроенный элемент комплексного XML-типа PROPERTY_Type. Его собственной моделью содержания является абстрактный комплексный XML-тип ANY_TYPE_type.

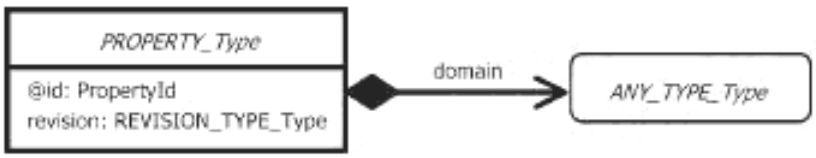


Рисунок 7 — UML-представление элемента XML-языка комплексного XML-типа данных

На рисунке 8 ниже дан текст программы на XML-языке, соответствующий рисунку 7.

```
<xs:complexType name="PROPERTY_Type" abstract="true">
    <xs:sequence>
        <xs:element name="revision" type="REVISION_TYPE_Type"/>
        <xs:element name="domain" type="ANY_TYPE_Type"/>
    </xs:sequence>
    <xs:attribute name="id" type="PropertyId" use="required"/>
</xs:complexType>
```

Рисунок 8 — XML-представление для XML-элемента комплексного типа

6.3.1.5 Представление кардинального числа встроенного элемента XML-языка

Кардинальное число элемента XML-языка описывается с помощью UML-обозначений: *minimum cardinality .. maximum cardinality*.

Пример 1 — На рисунке 9 минимальное кардинальное число равно 0, максимальное кардинальное число равно 1. Они назначены оператором is_deprecated и элементом XML-языка icon.

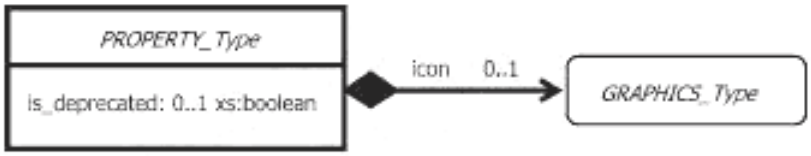


Рисунок 9 — UML-представление кардинального числа элемента XML-языка

Примечание 1 — Соотношение для кардинальных чисел представлено на рисунке 9 для иллюстрации функциональных возможностей.

Примечание 2 — Соглашение о цвете определено в 6.3.3.

Пример 2 — На рисунке 10 (ниже) дан текст программы на языке XML, соответствующий рисунку 9.

```
<xs:complexType name="PROPERTY_Type" abstract="true">
    <xs:sequence>
        <xs:element name="is_deprecated" type="xs:boolean" minOccurs="0"/>
        <xs:element name="icon" type="GRAPHICS_Type" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>
```

Рисунок 10 — XML-представление кардинального числа элемента XML-языка

6.3.1.6 Представление расширений комплексного XML-типа данных

Расширения комплексного XML-типа данных представлены обычным треугольником, характеризующим наследование.

Пример 1 — На рисунке 11 (ниже) комплексный XML-тип данных NON_DEPENDENT_P_DET_Type (независимый тип свойства) определен как расширение абстрактного комплексного XML-типа данных PROPERTY_Type.

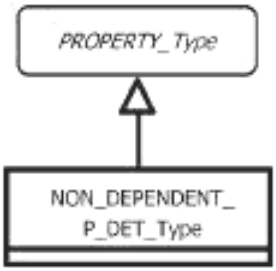


Рисунок 11 — UML-представление расширений комплексного XML-типа данных

Пример 2 — На рисунке 12 приведен текст программы на XML-языке, соответствующий рисунку 11.

```
<xs:complexType name="NON_DEPENDENT_P_DET_Type">
    <xs:complexContent>
        <xs:extension base="PROPERTY_Type"/>
    </xs:complexContent>
</xs:complexType>
```

Рисунок 12 — XML-представление для расширений комплексного XML-типа данных

6.3.2 Методика ссылок на понятия CIIM-онтологии

В данном разделе определены графические обозначения, используемые для идентификации понятий онтологии CIIM и для ссылок (на них) других понятий онтологии CIIM. Здесь также введены графические обозначения многократных ссылок одного понятия онтологии CIIM на множество других понятий онтологии CIIM.

6.3.2.1 Идентификация понятия онтологии CIIM

CIIM-модель дает порядок связи глобального идентификатора с понятием CIIM-онтологии.

В OntoML-языке для идентификации заданного типа понятия CIIM-онтологии устанавливается особый комплексный XML-тип данных:

- имена указанных комплексных XML-типов данных отражают имена соответствующих целевых типов понятий;
- каждый указанный комплексный XML-тип данных содержит атрибут, значение которого является глобальным идентификатором особого понятия CIIM-онтологии, которое данный атрибут идентифицирует;
- имя данного атрибута отражает также имя рассматриваемого целевого типа.

Такие элементы встраиваются внутрь понятий CIIM-онтологии.

Имена указанных комплексных XML-типов данных определяются по нижеследующему правилу:

- поставщик: для комплексного XML-типа данных **SUPPLIER_Type** тип идентификатора — **SupplierId**;
- класс: для комплексного XML-типа данных **CLASS_Type** тип идентификатора — **ClassId**;
- свойство: для комплексного XML-типа данных **PROPERTY_Type** тип идентификатора — **PropertyId**;
- тип данных: для элементов комплексного XML-типа данных **DATATYPE_type** тип идентификатора — **DatatypeId**;
- документ: для комплексного XML-типа данных **DOCUMENT_Type** тип идентификатора — **DocumentId**.

Примечание — Структура указанных типов идентификаторов определена в 9.1.

Имя данного атрибута идентификации **"id"**. В настоящем стандарте оно представлено в виде **@id**. Данное имя показывает, что оно является XML-атрибутом языка и не встроенным XML-элементом.

Пример — На рисунке 13 глобальный идентификатор понятия класса онтологий, представленный комплексным XML-типом данных CLASS_Type, содержит атрибут @id для типа данных ClassId.

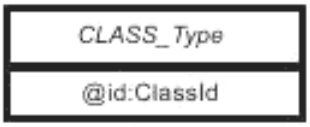


Рисунок 13 — Идентификация понятия CIIM-онтологии

6.3.2.2 OntoML-представление ссылки на понятие CIIM-онтологии

Для ссылки на заданный тип понятия CIIM-онтологии определяется особый комплексный XML-тип данных:

- имена указанных комплексных XML-типов данных отражают имя соответствующего целевого типа данных;
- каждый ссылочный элемент содержит атрибут, значение которого является глобальным идентификатором понятия CIIM-онтологии, на которое данный атрибут ссылается;
- имя данного атрибута также отражает имя рассматриваемого целевого типа.

Ссылочный комплексный XML-тип данных и имя ссылочного XML-атрибута (приставка «@» указывает, что это атрибут XML-языка) определены в соответствии со ссылочными понятиями онтологии CIIM:

- комплексный XML-тип данных **SUPPLIER_REFERENCE_Type** и атрибут XML-языка **supplier_ref** (с типом данных **SupplierId**): ссылка на поставщика;
- комплексный XML-тип данных **CLASS_REFERENCE_Type** и атрибут XML-языка **class_ref** (с типом данных **ClassId**): ссылка на класс;
- комплексный XML-тип данных **PROPERTY_REFERENCE_Type** и атрибут XML-языка **property_ref** (с типом данных **PropertyId**): ссылка на свойство;
- комплексный XML-тип данных **DATATYPE_REFERENCE_Type** и атрибут XML-языка **datatype_ref** (с типом данных **DatatypeId**): ссылка на тип данных;
- комплексный XML-тип данных **DOCUMENT_REFERENCE_Type** и атрибут XML-языка **document_ref** (с типом данных **DocumentId**): ссылка на документ.

Примечание — Тип ссылочного атрибута должен соответствовать типу ссылочного понятия CIIM-онтологии.

Пример — Ссылка на понятия класса онтологий производится с помощью комплексного XML-типа данных CLASS_REFERENCE_Type и ссылочного XML-атрибута @class_ref (с типом данных ClassId) (см. рисунок 14).

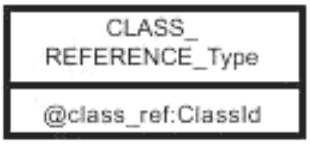


Рисунок 14 — Ссылка на понятие CIIM-онтологии

6.3.2.3 OntoML-представление для простых и многозначных ссылок на понятия CIIM-онтологии

Если ссылка одного понятия CIIM-онтологии на другое понятие CIIM-онтологии является многозначной, то создается дополнительный комплексный XML-тип данных (см. рисунок 15).

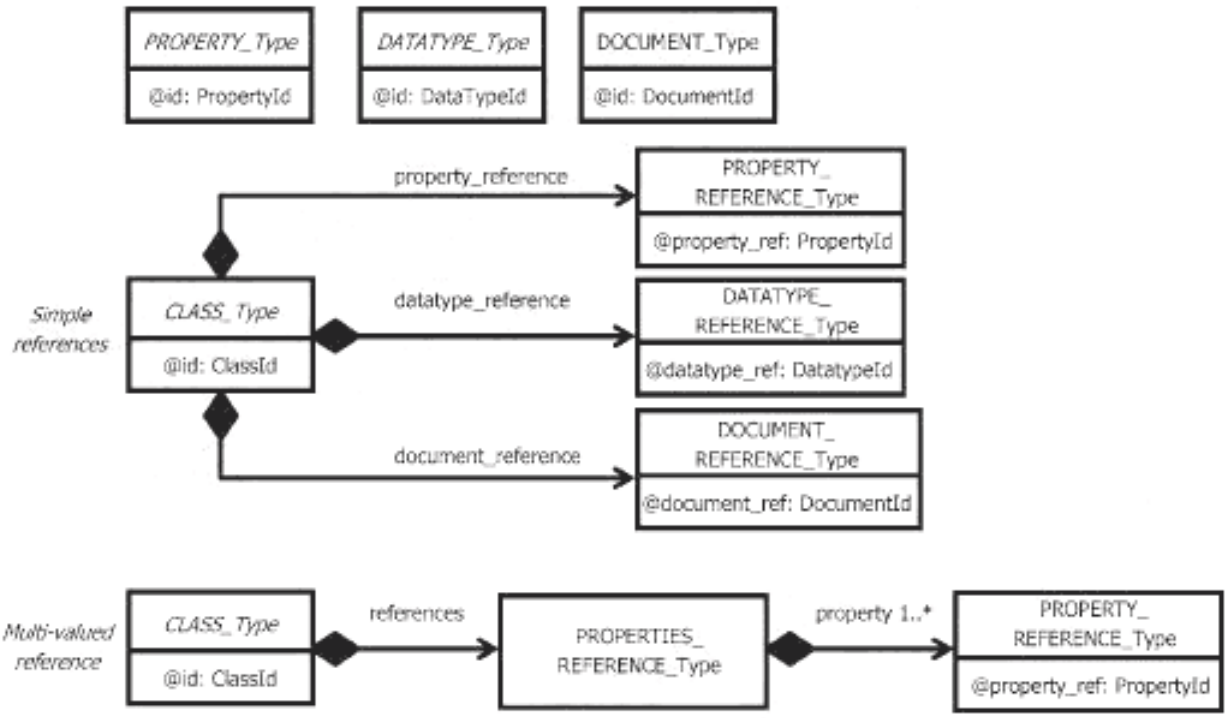


Рисунок 15 — Ссылка на понятие CIIM-онтологии

На данном рисунке:

- закрашенный ромб представляет составное соотношение для базовой вложенной XML-структуры;
- стрелка указывает соответствующую ориентацию соотношения.

На данном рисунке определены свойства, типы данных, документ и классы онтологии, представленные комплексными XML-типами данных **PROPERTY_Type, DATATYPE_type, DOCUMENT_Type** и **CLASS_Type** соответственно. Все они идентифицированы атрибутом XML-языка «**id**». Тип атрибута зависит от идентифицированного понятия CIIM-онтологии. Рассмотрены два варианта соотношений:

- *простая ссылка:* это соотношение между одним классом и одним свойством, типом данных или документом. Данное соотношение представлено XML-элементом (соответственно *property_reference, datatype_reference* и *document_reference*). Определение содержания: комплексные XML-типы данных **PROPERTY_REFERENCE_Type, DATATYPE_REFERENCE_Type** и **DOCUMENT_REFERENCE_Type** соответственно;

- *многозначная ссылка:* это соотношение между одним классом и несколькими свойствами. Данное соотношение представлено элементом XML-языка (*ссылками*). Оно работает по принципу контейнера. Определение содержания: комплексный XML-тип данных **PROPERTY_REFERENCE_Type**.

6.3.2.4 Упрощенное графическое представление ссылок на понятия CIIM-онтологии

В соответствии с 6.3.2.3 ссылки между понятиями CIIM-онтологии включают соответствующий идентификатор (ссылку отображения) и последующую комплексную цепочку одного-двух составных соотношений. Для упрощения графического представления вводится особое графическое обозначение. Ссылки между понятиями CIIM-онтологии представляются закрашенным ромбом и штриховой линией, соединяющей ссылочный комплексный XML-тип данных и ссылочное понятие CIIM-онтологии.

Целевое понятие CIIM-онтологии представляется соответствующим именем в штриховом прямоугольнике заглавными буквами:

- поставщик: **SUPPLIER**;
- класс: **CLASS**;
- свойство: **PROPERTY**;

- тип данных: **DATATYPE**;
- документ: **DOCUMENT**.

Если соотношение многозначно, то кардинальное число соотношения представляется как в UML-языке.

Пример 1 — На рисунке 16 дана простая ссылка между двумя понятиями CIIM-онтологии (вместе с ее значением) с помощью указанных выше обозначений.

Common notation	Полное обозначение
References	Ссылки
Simplified notation	Упрощенное обозначение

Рисунок 16 — UML-представление простой ссылки между понятиями CIIM-онтологии

Пример 2 — На рисунке 17 приведен текст программы на XML-языке, соответствующий рисунку 16.

```
<xs:complexType name="GLASS_Type">
    <xs:sequence>
        <xs:element name="references" type="PROPERTY_REFERENCE_Tupe"/>
    </xs:sequence>
</xs:complexType>

<xs:complexType name="PROPERTY_REFERENCE_Tupe">
    <xs:attribute name="property_ref" type="PropertyId" use="required"/>
</xs:complexType>
```

Рисунок 17 — Представление простой XML-ссылки между понятиями CIIM-онтологии

Пример 3 — На рисунке 18 представлена многозначная ссылка между понятиями CIIM-онтологии.

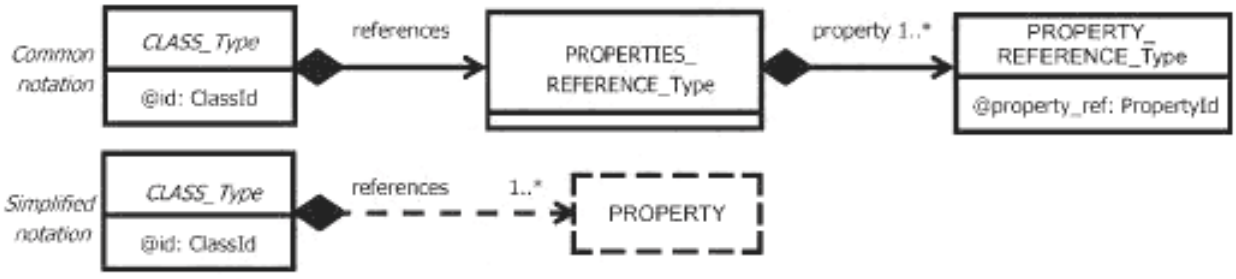


Рисунок 18 — UML-представление многозначной ссылки между понятиями CIIM-онтологии

Пример 4 — На рисунке 19 приведен текст программы на XML-языке, соответствующий рисунку 18.

```
<xs:complexType name="GLASS_Type">
    <xs:sequence>
        <xs:element name="references" type="PROPERTIES_REFERENCE_Tupe"/>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="PROPERTIES_REFERENCE_Type">
    <xs:sequence>
        <xs:element name="property"
            type="PROPERTY_REFERENCE_Type" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
<xs:complexType name="PROPERTY_REFERENCE_Type">
        <xs:attribute name="property_ref" tupe="PropertyId" use="required"/>
</xs:complexType>
```

Рисунок 19 — Представление многозначной XML-ссылки между понятиями CIIM-онтологии

6.3.3 Соглашение о цвете UML-диаграмм

Соглашение о цвете UML-диаграмм позволяет выделить на схеме атрибуты и элемент XML-языка, являющиеся обязательными для описания любого информационного элемента (понятия CIIM-онтологии, блоки информации). Суть соглашения:

- черные линии/текст, — если информационный элемент является обязательным;
- серые линии/текст, — если информационный элемент используется по выбору.

6.3.4 Описание структуры всех комплексных OntoML-типов

Далее структура и содержание всех комплексных OntoML-типов определяется в нескольких разделах. Каждый раздел фокусируется на одном особом комплексном XML-типе данных или, возможно, на небольшом количестве связанных комплексных XML-типов данных, не встроенных внутрь отдельных типов XML. Данный тип XML-данных называется основным типом раздела. Основные типы раздела четко идентифицируются своим именем и заголовком раздела.

Пример — Подраздел 6.6, озаглавленный "Корневой элемент онтологии". В заголовке написано: "В языке OntoML все блоки информации онтологии собраны в одну общую структуру — комплексный XML-тип данных DICTIONARY_TYPE". Таким образом, словарный тип DICTIONARY_TYPE является основным типом раздела.

Описание становится более целостным, если этот же раздел определяет содержание и структуру ряда других комплексных OntoML-типов, соединенных с основным типом (основными типами) раздела как по наследству, так и по композиции.

Далее приведено описание указанного набора комплексных типов.

6.3.4.1 Графические представления

В каждом разделе полная структура основного типа (основных типов) раздела определена графически с помощью UML-обозначений, представленных выше. Основной тип (основные типы) раздела может включать встроенные элементы XML-языка, модели содержания которых определены комплексными типами.

Некоторые из рассматриваемых комплексных XML-типов данных представлены прямоугольниками без скругления углов. Это означает, что вся структура и содержание указанных комплексных типов определены в настоящем разделе. Указанные структуры определены на одном и том же рисунке (см. рисунок со встроенным комплексным типом).

Пример 1 — На рисунке 21 HEADER_Type (тип заголовка) содержит элемент XML-языка ontoml_information, модель содержания которого определена комплексным XML-типом данных INFORMATION_Type. Данный тип представлен прямоугольником без скругления углов. Таким образом, его полная структура определена на рисунке 21. Он включает различные элементы XML-языка: synonnymous_name, PREFERRED_name, short_name, icon, remark и note.

Содержание указанного комплексного XML-типа данных определено под заголовком "Определение внутреннего типа" в том же разделе.

Пример 2 — В 6.5 содержание типа INFORMATION_Type определено в "Определение внутреннего типа" под заголовком: "Перечень описаний классов, содержащихся в словаре".

Некоторые другие комплексные XML-типы представлены скругленными прямоугольниками. Их структура и содержание определены в других разделах настоящего стандарта. Их количество определено под заголовком "Определение внешнего типа".

Пример 3 — В 6.5 модель содержания встроенного элемента XML-языка icon является комплексным XML-типом данных GRAPHICS_Type. Соответствующий прямоугольник на схеме имеет скругленные углы. Это означает, что данный комплексный тип определен в другом разделе. Под заголовком "Определение внешнего типа" подраздела 6.5 указано, что тип GRAPHICS_Type определен в 8.2.2.2.

6.3.4.2 Определение внутреннего элемента

Под заголовком "Определение внутреннего элемента" в каждом разделе определяются все атрибуты XML-языка и все элементы XML-языка, встроенные внутрь комплексных XML-типов данных, определенных в данном разделе. Составлен список указанных элементов (атрибутов XML-языка и встроенных элементов XML-языка, принадлежащих основному типу раздела). Указанные элементы (принадлежащие встроенному элементу XML-языка, чей комплексный тип представлен прямоугольниками без скругления углов) идентифицированы по обозначению маршрута, начиная с основного элемента. Разделителем маршрута является наклонная черта ('/').

Пример 1 — Под заголовком «Определение внутреннего элемента» подраздела 6.6 определение элемента XML-языка class (встроенное внутрь элемента contained_classes типа данных DICTIONARY_Type) ассоциируется с нижеследующим идентификатором: contained_classes/class.

Если данный раздел обращается к нескольким основным типам раздела, соединенным наследственным соотношением, то указанные элементы, принадлежащие корню наследственной иерархии, не классифицируются. Данные элементы, принадлежащие дочерним классам, классифицируются по имени класса (в скобках).

Пример 2 — 5.7.3 определяет простые свойства уровня онтологии, включающие тип данных PROPERTY_Type, независимый тип данных NON_DEPENDENT_P_DET_Type, условный тип данных CONDITION_DET_Type и зависимый тип данных DEPENDENT_P_DET_Type. Под заголовком "Определение внутреннего элемента" в данном пункте определение элемента языка XML depends_on (зависит от) (встроенного внутрь подтипа типа данных DEPENDENT_P_DET_Type для типа данных PROPERTY_Type) ассоциируется с нижеследующим идентификатором: depends_on (DEPENDENT_P_DET_Type).

6.3.4.3 Определение внутренних типов данных

Под данным заголовком определяется содержание:

- типов атрибутов комплексных XML-типов данных, определенных в рассматриваемом разделе;
- типов XML-элементов, которые вводятся в один из комплексных XML-типов данных, определенных в рассматриваемом разделе и того XML-типа, который является простым;
- типов XML-элементов, которые вводятся в один из комплексных XML-типов данных, определенных в рассматриваемом разделе и того XML-типа данных, который является комплексным XML-типом, выражаемым цифрами в прямоугольниках.

6.3.4.4 Определение внешнего типа данных

Под данным заголовком ниже будет определяться каждый комплексный XML-тип данных, представляемый на рисунке в виде прямоугольника с закругленными углами, связанного с номером раздела, где они были определены.

6.3.4.5 Перечень ограничительных условий

Под данным заголовком ниже будут определяться дополнительные ограничительные условия, которые не могут быть представлены с помощью унифицированного языка моделирования (UML).

6.4 Общая структура OntoML-языка

Совместимый с OntoML-языком экземпляр XML-документа позволяет представлять данные, описывающие онтологию, экземпляры или и то и другое. Верхний уровень экземпляра OntoML-документа определяется посредством комплексного XML-типа данных **ONTOML_Type** (см. рисунок 20).

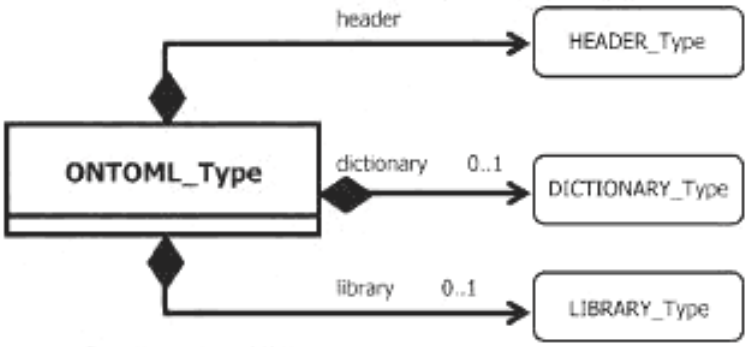


Рисунок 20 — UML-диаграмма структуры онтологии

Определения внутренних элементов:

Элемент **dictionary:** Определяет понятия CIIM-онтологии, которые составляют обмениваемую онтологию.

Элемент **header**: Определяет общую информацию относительно обмениваемого файла.

Элемент **library**: Определяет множество описаний продукции, которые составляют содержание обмениваемой библиотеки.

Определения внешних типов:

Тип **DICTIONARY_Type:** Является спецификацией OntoML-онтологии, см. 6.6.

Тип **HEADER_Type:** Является спецификацией заголовка OntoML XML-документа, см. 6.5.

Тип **LIBRARY_Type:** Является спецификацией OntoML-библиотеки, см. 7.

Перечень ограничительных условий:

Существует либо XML-элемент словаря, либо XML-элемент библиотеки, либо они существуют оба.

6.5 Заголовок в OntoML-языке

Заголовок в OntoML-языке предоставляет версию настоящего стандарта для создания экземпляра OntoML-документа и воспринимаемой человеком информации относительно этого экземпляра. Кроме того, он дает информацию об общей структуре экземпляра OntoML-документа и представляется с помощью комплексного XML-типа данных **HEADER_Type** (см. рисунок 21).

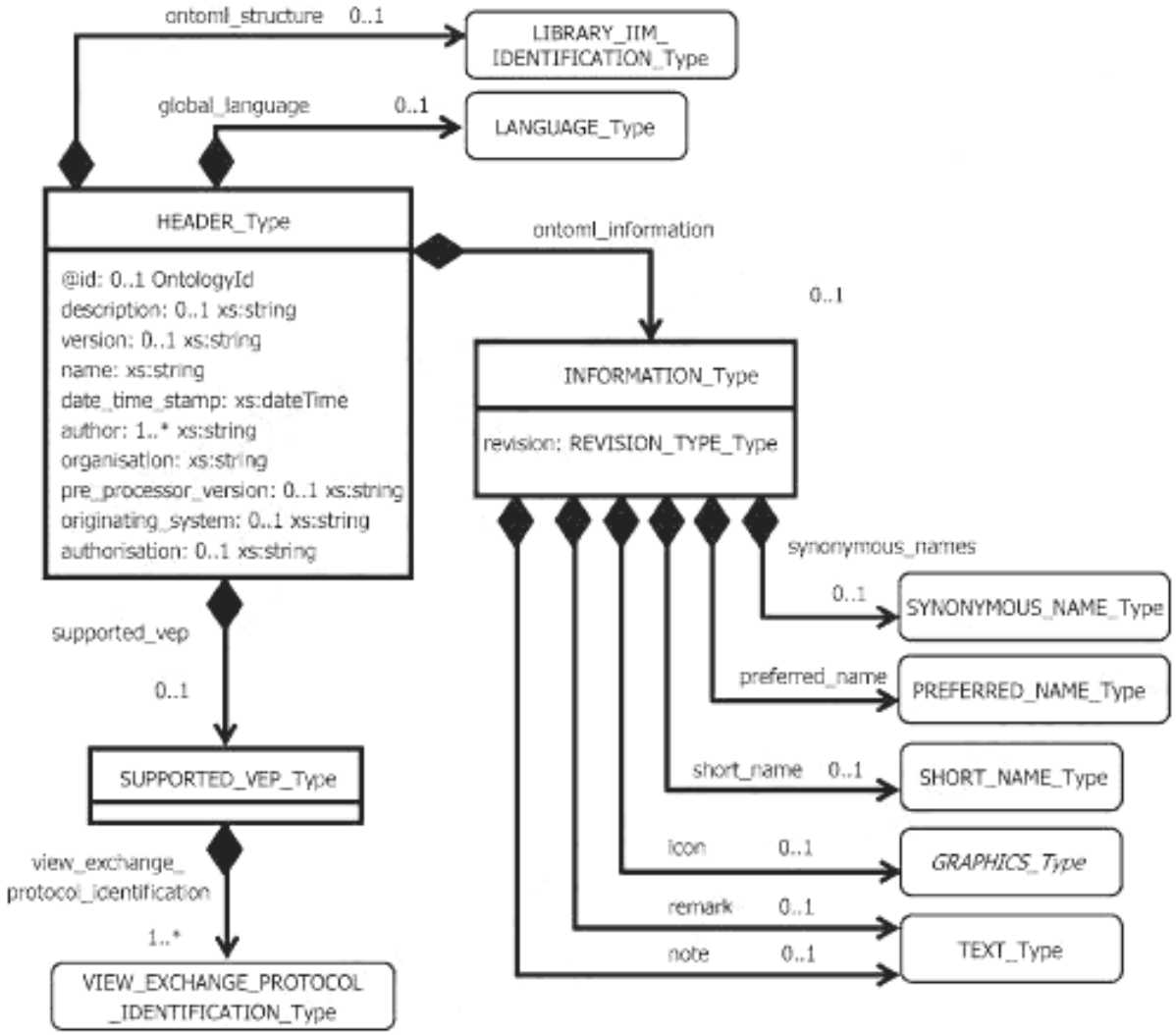


Рисунок 21 — Структура заголовка онтологии

Определения внутренних элементов:

Элемент **@id:** Определяет возможный идентификатор словаря, к которому принадлежит определенный класс.

Элемент **author:** Определяет имя и почтовый адрес лиц, которые ответственны за создание структуры обмена.

Элемент **authorisation:** Определяет имя и почтовый адрес лица, который уполномочен на посылку структуры обмена.

Элемент **date_time_stamp:** Определяет дату и время в случае, если структура обмена была создана.

Элемент **description:** Определяет неформальное описание содержания экземпляра OntoML-документа.

Элемент **global_language:** Определяет возможный глобальный язык, используемый для описания непереведенной информации, связанной с любым понятием CIIM-онтологии.

Элемент **name:** Определяет строку, используемую для наименования конкретного экземпляра OntoML-документа.

Элемент **ontoml_information/icon:** Определяет дополнительные графические материалы, которые представляют описание, связанное с различными именами, предоставляемыми для описания OntoML-словаря и/или библиотеки.

Элемент **ontoml_information/note:** Определяет дополнительную информацию к любой части словаря и/или библиотеки, которая существенна для понимания этой информации.

Элемент **ontoml_information/preferred_name:** Определяет имя словаря и/или библиотеки, которое предпочтительно для использования.

Элемент **ontoml_information/remark:** Определяет пояснительный текст, дополнительно поясняет содержание данного словаря и/или библиотеки.

Элемент **ontoml_information/revision:** Определяет номер редакции словаря и/или библиотеки.

Элемент **ontoml_information/short_name:** Определяет сокращение предпочтительного имени.

Элемент **ontoml_information/synonymous_names:** Определяет совокупность синонимических имен.

Элемент **ontoml_structure:** Определяет встроенную в библиотеку информационную модель, которая реализует OntoML-словарь и/или библиотеку.

Элемент **organisation:** Определяет группу или организацию, которые ответственны за онтологическую структуру обмена/экземпляр документа.

Элемент **originating_system:** Определяет систему, из которой исходит структура обмена данными.

Элемент **pre_processor_version:** Определяет систему, используемую для создания структуры обмена данными, включая системное имя продукции и ее вариант.

Элемент **revision:** Определяет редакцию OntoML-диаграммы, с которой совпадает структура обмена данными.

Элемент **supported_vep:** Определяет список видов протоколов обмена данными, поддерживаемых словарем и/или библиотекой.

Элемент **supported_vep/view_exchange_protocol_identification:** Определяет вид протокола обмена данными, поддерживаемого словарем и/или библиотекой.

Элемент **version:** Определяет версию OntoML-диаграммы, которой соответствует структура обмена данными.

Определения внутренних типов:

Тип **INFORMATION_Type:** Является незашифрованной текстовой информацией (возможно, переведенной), предоставленной словарем и/или библиотекой.

Тип **REVISION_TYPE_Type:** Является строкой (**xs:string** XML-диаграммы), которая представляет собой значения, допускаемые для проверки и имеет длину, не превышающую 3 символа.

Тип **SUPPORTED_VEP_Type:** Является спецификацией видов протоколов обмена данными, поддерживаемых словарем и/или библиотекой.

Определения внешних типов:

Тип **OntologyId:** См. 9.1.

Тип **GRAPHICS_Type:** См. 8.2.2.2.

Тип **LANGUAGE_Type:** См. 8.1.1.

Тип **LIBRARY_IIM_IDENTIFICATION_Type:** См. 8.7.1.

Тип **PREFERRED_NAME_Type:** См. 8.1.2.

Тип **SHORT_NAME_Type:** См. 8.1.2.

Тип **SYNONYMOUS_NAME_Type:** См. 8.1.2.

Тип **TEXT_Type:** См. 8.1.2.

Тип **VIEW_EXCHANGE_PROTOCOL_IDENTIFICATION_Type:** См. 8.7.2.

6.6 Корневой элемент онтологии

В OntoML-языке каждый фрагмент онтологии информации объединяется в общую структуру, которая принадлежит комплексному XML-типу данных **DICTIONARY_TYPE** (см. рисунок 22).

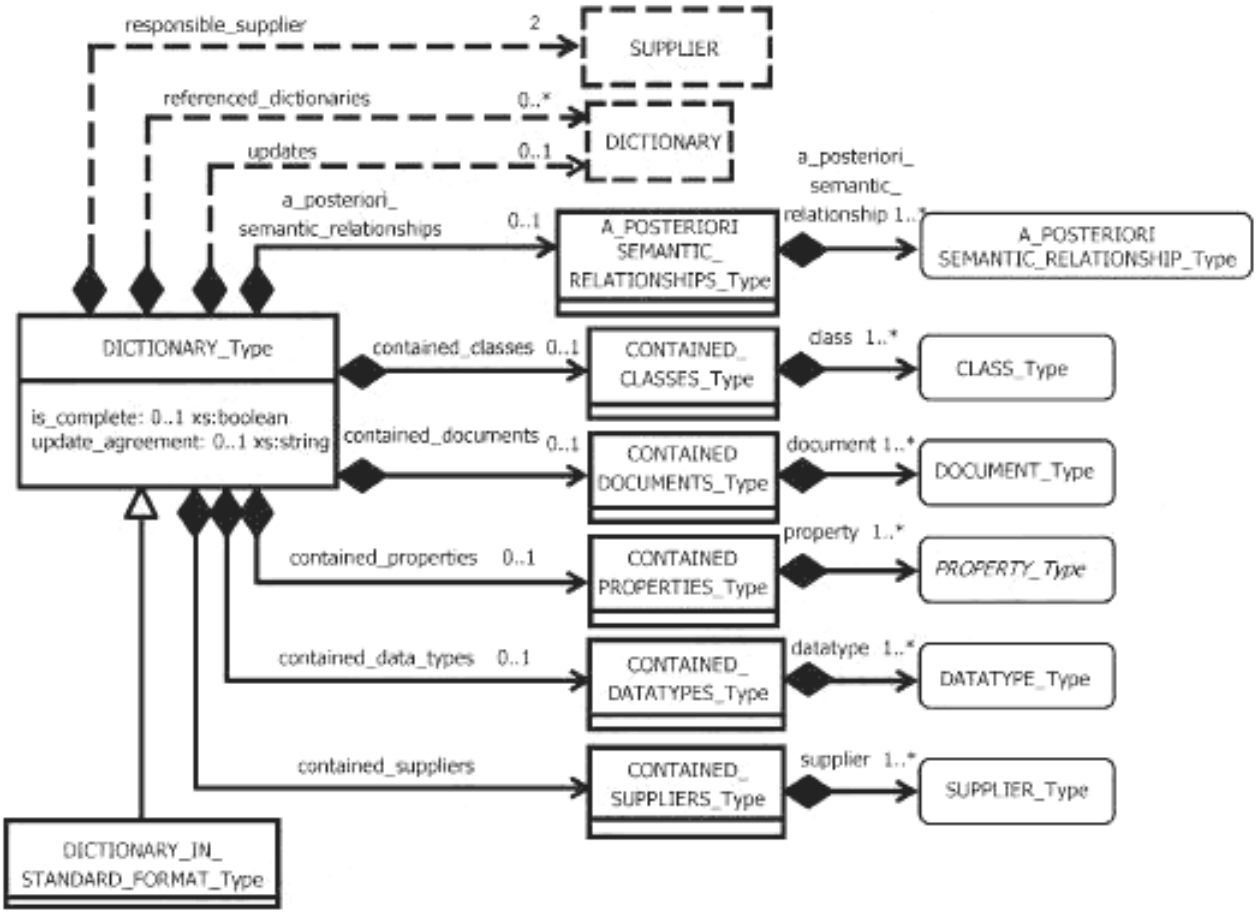


Рисунок 22 — Корневой элемент онтологии

Определения внутренних элементов:

Элемент **a_posteriori_semantic_relationships:** Определяет список *апостериорных* связей, содержащихся в словаре.

Элемент **a_posteriori_semantic_relationships/a_posteriori_semantic_relationship:** Определяет *апостериорную* связь, содержащуюся в словаре.

Элемент **contained_classes:** Определяет список описаний классов, содержащихся в словаре.

Элемент **contained_classes/class:** Определяет описание класса, содержащееся в словаре.

Элемент **contained_datatypes:** Определяет список описаний типов данных, содержащихся в словаре.

Элемент **contained_datatypes/datatype:** Определяет описание типа данных, содержащееся в словаре.

Элемент **contained_documents:** Определяет список описаний документа, содержащихся в словаре.

Элемент **contained_documents/document:** Определяет описание документа, содержащееся в словаре.

Элемент **contained_properties:** Определяет список описаний свойства, содержащихся в словаре.

Элемент **contained_properties/property:** Определяет описание свойства, содержащееся в словаре.

Элемент **contained_suppliers:** Определяет список описаний поставщиков, содержащихся в словаре.

Элемент **contained_suppliers/supplier:** Определяет описание поставщика, содержащееся в словаре.

Элемент **is_complete:** Определяет, полностью описывает ли словарь обмениваемую онтологию или только ее изменения.

Примечание 1 — XML-элемент **is_complete** используется только в том случае, когда словарь идентифицируется посредством его XML-атрибута **@id.**

Элемент **referenced_dictionaries:** Определяет идентификаторы словаря (при их наличии), указывая ссылки на другие словари, на которые в данном словаре ссылаются некоторые классы.

Элемент **responsible_supplier:** Определяет возможного поставщика данных, ответственного за понятия онтологии.

Примечание 2 — На поставщика всего словаря или частей его содержания ссылаются как на элемент **responsible_supplier** только тогда, когда он отвечает за экземпляр OntoML-документа. Кроме того, на него ссылаются в XML-элементе **contained_supplier**.

Элемент **update_agreement:** Определяет идентификатор (при его наличии), устанавливающий процесс, который будет использоваться для создания словаря на приемной системе (из списка словарей, определенных в XML-элементе **updates**). Элемент **update_agreement** может использоваться только тогда, когда XML-элемент **updates** будет использоваться сам по себе.

Элемент **updates:** Определяет идентификацию словаря (при его наличии) из предположения, что он уже доступен на приемной системе и способен создать полное содержание этого словаря.

Примечание 3 — XML-элемент **updates** может существовать только тогда, когда существует XML-элемент **identified_by**, а также тогда, когда XML-элемент **is_compelete** определен как ошибочный.

Определения внутренних типов:

Тип **CONTAINED_CLASSES_Type:** Является последовательностью описаний классов.

Тип **CONTAINED_DATATYPES_Type:** Является последовательностью описаний типов данных.

Тип **CONTAINED_DOCUMENTS_Type:** Является последовательностью описаний документов.

Тип **CONTAINED_PROPERTIES_Type:** Является последовательностью описаний свойств.

Тип **CONTAINED_SUPPLIERS_Type:** Является последовательностью описаний поставщиков.

Тип **DICTIONARY_IN_STANDARD_FORMAT_Type:** Является словарем, в котором используются только протоколы внешних файлов, допускаемые либо встроенной в библиотеку информационной моделью (индицируемой XML-элементом **library_structure**), либо протоколами просмотра обмениваемыми данными, на которые имеется ссылка XML-элемента **supported_vep** (оба элемента определены в комплексном XML-типе **HEADER_Type**).

Определения внешних типов:

Тип **A_POSTERIORI_SEMANTIC_RELATIONSHIP_Type:** См. 8.6.

Тип **CLASS_Type:** Является описанием классов словаря, см. 6.7.2.

Тип **DATATYPE_Type:** Является описанием типов данных словаря, см. 6.7.6.

Тип **DOCUMENT_Type:** Является описанием документов словаря, см. 6.7.7.

Тип **PROPERTY_Type:** Является описанием свойств словаря, см. 6.7.4.

Тип **SUPPLIER_Type:** Является описанием поставщиков словаря, см. 6.7.1.

Перечень ограничительных условий:

Если заголовочная часть обменного OntoML-файла содержит онтологический идентификатор продукции (XML-атрибут **id** комплексного XML-типа данных **HEADER_Type**), то должен предоставляться и информационный XML-элемент **is_complete**.

Если заголовочная часть обменного OntoML-файла содержит онтологический идентификатор продукции (XML-атрибут **id** комплексного XML-типа данных **HEADER_Type**), то ссылочный элемент **responsible_supplier** должен быть аналогичным идентифицированному поставщиком в онтологическом идентификаторе.

Элемент **updates** не должен существовать, если онтология продукции не идентифицирована (XML-атрибут **id** комплексного XML-типа данных **HEADER_Type**) или если XML-элемент **is_complete** установлен как истинный.

Если онтология продукции идентифицирована (XML-атрибут **id** комплексного XML-типа **HEADER_Type**), а также определен элемент **updates,** то идентифицированная онтология продукции должна иметь такой же код и того же поставщика, что и в ссылочном XML-элементе **updates,** и должна иметь более позднюю версию, чем в ссылочном элементе **updates** словаря.

6.7 OntoML-представление понятий CIIM-онтологии

В данном разделе определено OntoML-представление различных понятий CIIM-онтологии.

6.7.1 Поставщик

Онтологическое понятие поставщика связано с описанием организации, ответственной за определенную информацию, которая идентифицирована в экземпляре OntoML-документа. Это понятие иллюстрируется нижеприведенной UML-диаграммой (см. рисунок 23).

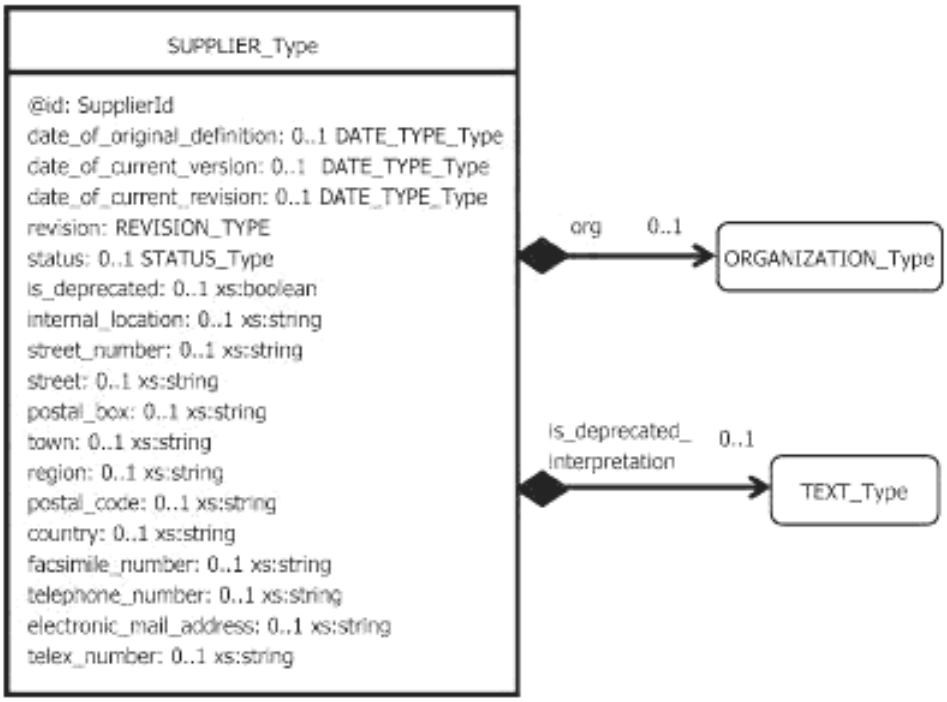


Рисунок 23 — UML-диаграмма онтологии поставщика

Определения внутренних элементов:

Элемент **@id:** Определяет идентификатор поставщика.

Элемент **country:** Определяет имя страны.

Элемент **date_of_original_definition:** Определяет дату, связанную с первым неизменным вариантом определения поставщика.

Элемент **date_of_current_version:** Определяет дату, связанную с существующим вариантом определения поставщика.

Элемент **date_of_current_revision:** Определяет дату, связанную с текущей проверкой определения поставщика.

Элемент **electronic_mail_address:** Определяет электронный адрес, по которому может получаться электронная почта.

Элемент **facsimile_number:** Определяет номер, по которому может получаться факс.

Элемент **internal_location:** Определяет задаваемый организацией адрес для внутренней системы связи.

Элемент **is_deprecated:** Определяет булев элемент, который (если он истинен) будет означать, что определение поставщика больше не используется.

Элемент **is_deprecated_interpretation:** Определяет способ интерпретации причины возражения, примерные значения от возражающего поставщика и соответствующий идентификатор.

Элемент **org:** Определяет организационную информацию поставщика.

Элемент **postal_box:** Определяет номер почтового ящика.

Элемент **postal_code:** Определяет код, который используется почтовой службой страны.

Элемент **region:** Определяет наименование региона.

Элемент **revision:** Определяет номер редакции существующего определения поставщика.

Элемент **status:** Определяет состояние в жизненном цикле определения поставщика.

Примечание 1 — Допустимые значения элемента **status** определяются путем заключения частного соглашения между поставщиком словаря и его пользователями.

Примечание 2 — Если XML-элемент status не предоставляется и если определение поставщика не вызывает возражений и может обозначаться с помощью XML-элемента **is_deprecated,** то определение поставщика должно иметь то же состояние стандартизации, что и у всей онтологии, в которой этот элемент используется. В частности, если онтология стандартизирована, то определение поставщика будет являться частью текущей редакции стандарта.

Элемент **street:** Определяет название улицы.

Элемент **street_number:** Определяет номер дома на улице.

Элемент **telephone_number:** Определяет номер телефона, по которому можно получить телефонный вызов.

Элемент **telex_number:** Определяет номер факса, по которому можно получать сообщения.

Элемент **town:** Определяет название города.

Определения внутренних типов:

Тип **DATE_TYPE_Type:** Является идентификатором значений, допустимых для даты (специальный тип данных xs:date XML-диаграммы).

Тип **REVISION_TYPE_Type:** Является строкой (**xs:string** XML-диаграммы), которая представляет собой значения, допускаемые для проверки. Эта строка должна содержать не более трех символов.

Тип **STATUS_Type:** Является строкой (**xs:string** XML-диаграммы), которая представляет собой значения, допускаемые для определения состояния. Эта строка не должна содержать дефисов «-» или пробелов.

Определения внешних типов:

Элемент **SupplierId:** См. 9.1.

Тип **ORGANIZATION_Type:** См. 8.8.1.

Перечень ограничительных условий:

Элементам **internal_location, street_number, street, postal_box, town, region, postal_code, country, facsimile_number, telephone_number, electronic_mail_address** или **telex_number** необходимо присвоить значение.

Если элемент **is_deprecated** существует, то должен существовать и элемент **is_deprecated_interpretation**.

Примерные значения элемента **is_deprecated_interpretation** должны определяться в тот момент, когда принималось решение о возражении.

6.7.2 Простой класс онтологии

В OntoML-языке определены три подтипа обобщенного и абстрактного понятия простых классов:

- *Класс элементов* — позволяет определять характеристики любого вида элементов и в частности — особой продукции с помощью принадлежащего класса и набора пар «свойство/значение». Классы элементов, принадлежащие экземплярной (*is-a*) иерархии, связанной с наследованием свойств;

- *Класс категоризации* — позволяет классифицировать элемент, характеристики которого определены в классе элементов в различных системах классификации. Подобная классификация не предполагает наличия никаких дополнительных свойств;

- *Класс элементов case-of* — специальный вид класса элементов, который помимо наследуемых от экземплярного родителя (is-a parent) возможных свойств, заимствует также и некоторые свойства из некоторых других существующих классов, которые содержат этот класс в соответствии со своей областью применения.

Примечание — Класс характеризации продукции и класс категоризации определен в разделе 5 ИСО/МЭК 77-2:2008.

6.7.2.1 Класс элементов

Рисунок 24 иллюстрирует структуру класса элементов.

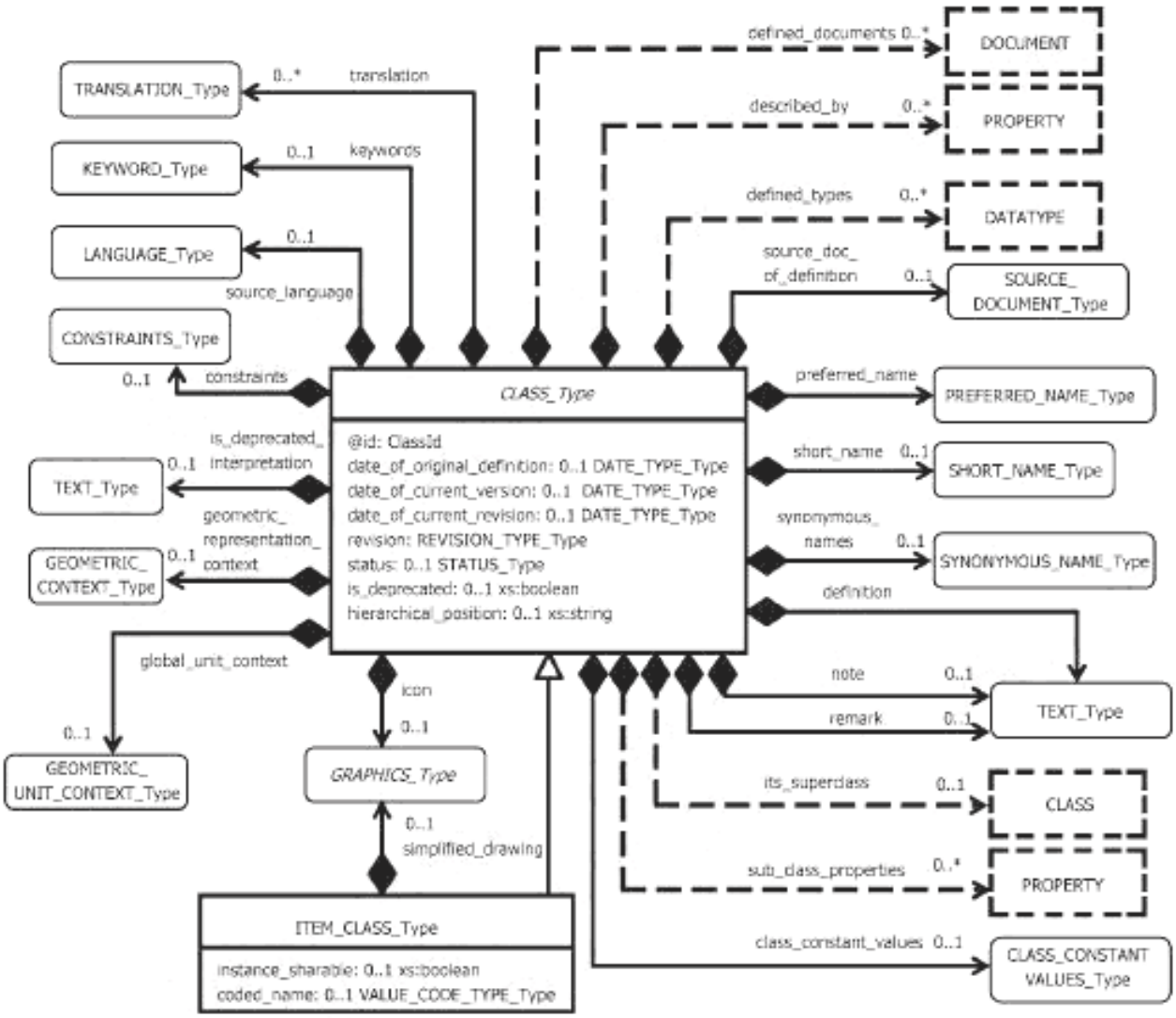


Рисунок 24 — UML-диаграмма, иллюстрирующая понятие простого класса онтологии

Класс элементов (представляемый с помощью комплексного XML-типа данных **ITEM_CLASS_Type**) наследует XML-описание содержания, определяемое абстрактным комплексным XML-типом данных **CLASS_Type**.

Примечание 1 — Наиболее общее представление классов требует использования только идентификатора (**@id**), элементов **revision**, **preferred_name** и **definition**.

Определения внутренних элементов:

Элемент **@id:** Определяет идентификатор класса.

Элемент **class_constant_values:** Определяет назначения элементов в текущем классе для значимых для класса свойств, признанных в суперклассах.

Примечание 2 — Элемент **class_constant_values** определяет селекторы классов, как это определено в разделе 5.5 Руководства ИСО/МЭК 77-2:2008. Элемент **coded_name** (типа **ITEM_CLASS_Type**) — это возможное закодированное имя класса.

Элемент **constraints:** Определяет набор условий, которые ограничивают целевые области значений некоторых «видимых» свойств класса некоторыми подклассами наследуемых областей значений.

Примечание 3 — Каждое ограничительное условие в совокупности этих условий должно выполняться с помощью экземпляров класса, поэтому эта совокупность является объединением ограничительных условий.

Элемент **date_of_current_revision:** Определяет дату, связанную с текущей проверкой определения класса.

Элемент **date_of_current_version:** Определяет дату, связанную с существующим вариантом определения класса.

Элемент **date_of_original_definition:** Определяет дату, связанную с первым неизменным вариантом определения класса.

Элемент **definition:** Определяет текст, устанавливающий данный класс.

Элемент **defined_documents:** Определяет набор ссылок на дополнительные документы, которые могут использоваться в пределах древа наследования свойств, происходящих из этого класса.

Примечание 4 — Каждый документ, на который ссылаются в совокупности элементов **defined_documents**, считается применимым в классе.

Элемент **defined_types:** Определяет набор ссылок на дополнительные типы, которые могут использоваться для различных свойств в пределах древа наследования свойств, происходящих из этого класса.

Примечание 5 — Каждый тип данных, на который ссылаются в совокупности элементов **defined_types**, считается применимым в классе.

Элемент **described_by:** Определяет список ссылок на дополнительные свойства, доступные к применению в описании экземпляров данного класса и любого из его подклассов (здесь и далее по тексту понятия экземпляр и экземпляр класса являются синонимами).

Примечание 6 — Каждое свойство, на которое ссылаются в совокупности элементов **described_by**, считается применимым в классе.

Примечание 7 — Свойство также может применяться к классу, если оно импортируется из другого класса посредством класса типа **ITEM_CLASS_CASE_OF_Type** (см. 6.7.2.3), типа **FUNCTIONAL_MODEL_CLASS_Type** (см. 6.7.3.2) или типа **FM_CLASS_VIEW_OF_Type** (см. 6.7.3.3). Поэтому свойства, на которые приводится ссылка с помощью атрибута **described_by**, не определяют все применимые свойства для этого класса.

Примечание 8 — Список порядка является представлением порядка следования свойств, предлагаемого поставщиком.

Элемент **geometric_representation_context:** Определяет эталонную систему координат для каждого свойства в классе, чей тип данных является позиционирующей единицей STEP, т. е. либо типа **PLACEMENT_TYPE_Type**, **AXIS1_PLACEMENT_TYPE_Type**, **AXIS2_PLACEMENT_2D_TYPE_Type** или **AXIS2_PLACEMENT_3D_TYPE_Type**.

Примечание 9 — Позиционирующие единицы STEP определены в разделе E1 Приложения E.

Примечание 10 — Определение местоположения эталонной системы координат по отношению к объекту, определенному своим классом, неформально описывается в элементе описания **geometric_representation_context**.

Пример 1 — Рассмотрим класс элементов, который описывает стеллажи, чьи передние панели являются прямоугольными. Поставщик желает определить путем размещения восьми вершин упаковочную коробку для каждого из стеллажей. Смысл геометрического представления, позволяющего определить эти вершины, может быть выражен следующим описанием: «Начало эталонной системы координат является точкой пересечения двух диагоналей на верхней поверхности стеллажа, при оси z, направленной вверх, оси x — направленной горизонтально в переднем направлении».

Примечание 11 — Элемент **geometric_representation_context** является OntoML-представлением элемента **geometric_representation_context**, определенного в ИСО 10303-42 для его геометрического представления. В ИСО 10303-42 данный контекст применим ко всем позиционирующим единицам STEP. В OntoML-языке данный контекст применим ко всем позиционирующим единицам, которые являются значениями свойств для класса, где определен элемент **geometric_representation_context**.

Элемент **global_unit_context:** Определяет единицу длины и, возможно, единицу угла, которые закреплены за контекстом геометрического представления всех позиционирующих единиц STEP-класса.

Примечание 12 — Элемент **global_unit_context** является OntoML-представлением элемента **global_unit_assigned_context**, определенного в ИСО 10303-42 для его геометрического представления. В ИСО 10303-42 данный контекст применим ко всем элементам геометрического представления. В OntoML-языке данный контекст применим ко всем позиционирующим единицам STEP, которые являются значениями свойств для класса, где определен элемент **global_unit_context**.

Примечание 13 — Если элемент **global_unit_context** не предоставляется, то значением единицы длины по умолчанию является миллиметр, а для единицы угла — градус.

Элемент **hierarchical_position:** Определяет кодированное представление положения класса в иерархии классов, к которой этот класс принадлежит.

Примечание 14 — Этот вид кодированного имени используется, в частности, в иерархиях категорий продукции для представления инклюзивной структуры классов посредством некоторого соглашения по кодированию.

Пример 2 — В UNSPSC-классификаторе промышленные компоненты и вспомогательные материалы имеют иерархическую позицию 31000000, аппаратные средства — иерархическую позицию 31160000, а болты — иерархическую позицию 31161600. Путем заключения соглашения это представление иерархических позиций позволяет считать промышленные компоненты и вспомогательные материалы первым уровнем иерархии, аппаратные средства — вторым уровнем, входящим в первый уровень, а болты — третьим уровнем, входящим во второй уровень.

Примечание 15 — Элемент **hierarchical_position** класса изменяется при изменении структуры класса онтологии, поэтому он не может использоваться в качестве неизменного идентификатора класса.

Элемент **icon:** Определяет графические материалы, представляющие собой описание, связанное с их именами.

Элемент **instance_sharable (ITEM_CLASS_Type):** Если этот элемент ошибочный, то он определяет то, что экземпляры класса элементов являются характерными; если этот элемент не предоставляется или является истинным, то он определяет то, что экземпляры класса элементов являются индивидуальными.

Примечание 16 — Общая модель словаря, определенная в ИСО 13584/МЭК 61360, является реализацией, зависящей от решения о том, должны ли представляться некоторые элементы реальных слов, моделируемые с помощью одного и того же набора пар «свойство/значение», в файле обмена данными с помощью нескольких конструкций описаний XML-элемента или с помощью той же конструкции описания XML-элемента. Таким образом, одиночная конструкция описания XML-элемента, чей элемент **instance_sharable** является ошибочным и на который дается ссылка с помощью нескольких конструкций описания элемента на уровне модели данных, интерпретируется как представление нескольких элементов реальных слов.

Элемент **is_deprecated:** Определяет булев элемент, который (если он истинен) будет означать, что определение поставщика больше не используется.

Элемент **is_deprecated_interpretation:** Определяет способ интерпретации причины исключения[1)], примерные значения от поставщика и соответствующий идентификатор.

Элемент **its_superclass:** Определяет ссылку на класс, текущий класс которого является подклассом.

Элемент **keywords:** Определяет набор ключевых слов (возможно, на нескольких языках), который позволяет находить нужный класс.

Элемент **note:** Определяет дополнительную информацию (возможно, переведенную) в любой части класса, которая существенна для ее понимания.

Элемент **preferred_name:** Определяет имя класса (возможно, переведенное), которое при его использовании можно считать преимущественным.

Элемент **remark:** Определяет пояснительный текст (возможно, переведенный), дополнительно проясняющий содержание данного класса.

Элемент **revision:** Определяет номер редакции существующего определения класса.

Элемент **short_name:** Определяет сокращение предпочтительного имени класса (возможно, переведенного).

Элемент **simplified_drawing:** Определяет чертеж, который может быть связан с описываемым классом.

Элемент **source_doc_of_definition:** Определяет возможный документ-источник, из которого заимствовано данное определение.

Элемент **source_language:** Определяет язык, на котором первоначально было дано описание класса и на котором сохраняется исходное содержание в случае несоответствия перевода.

Элемент **status:** Определяет состояние в жизненном цикле определения класса.

Примечание 17 — Допустимые значения элемента **status** определяются путем заключения частного соглашения между поставщиком словаря и его пользователями.

Примечание 18 — Если XML-элемент **status** не предоставляется и если это определение класса не вызывает возражений и может обозначаться с помощью XML-элемента **is_deprecated,** то определение класса должно иметь то же состояние стандартизации, что и у всей онтологии, в которой этот элемент используется. В частности, если онтология стандартизирована, то определение класса будет являться частью текущей редакции стандарта.

[1)] Может относиться к классу, интерфейсу, конструктору, методу или полю, использование которых больше не рекомендуется, так как они могут уже не существовать в будущей версии языка.

Элемент **sub_class_properties:** Определяет свойства, значимые для данного класса, т. е. в подклассах одно-единственное значение будет присваиваться каждому классу.

Примечание 19 — Элемент **sub_class_properties** определяет селекторы класса, как это определено в разделе 5.5 ИСО/МЭК 77-2:2008.

Элемент **synonymous_names:** Определяет совокупность синонимических имен предпочтительному имени (возможно, переведенному).

Элемент **translation:** Определяет возможную совокупность переведенной информации, предоставляемой для переводимых элементов.

Определения внутренних типов:

Тип **DATE_TYPE_Type:** Является значениями, допускаемыми для обозначения даты (специальный тип данных **xs:date** XML-диаграммы).

Тип **REVISION_TYPE_Type:** Является строкой (**xs:string** XML-диаграммы), которая представляет значения, допускаемые для обозначения редакции. Эта строка должна содержать не более трех символов.

Тип **STATUS_Type:** Является строкой (**xs:string** XML-диаграммы), которая представляет значения, допускаемые для обозначения состояния.

Тип **VALUE_CODE_TYPE_Type:** Является строкой (**xs:string** XML-диаграммы), которая представляет значения, допускаемые для обозначения кода. Эта строка должна содержать не более 35 символов.

Определения внешних типов:

Элемент **ClassId:** См. 9.1.

Элемент **ConstraintId:** См. 9.1.

Тип **CLASS_CONSTANT_VALUES_Type:** См. 6.7.2.4.

Тип **CONSTRAINT_Type:** См. 8.5.

Тип **GEOMETRIC_CONTEXT:** См. 8.8.3.

Тип **GEOMETRIC_UNIT_CONTEXT:** См. 8.8.4.

Тип **GRAPHICS_Type:** См. 8.2.2.2.

Тип **KEYWORD_Type:** См. 8.1.2.

Тип **LANGUAGE_Type:** См. 8.1.1.

Тип **PREFERRED_NAME_Type:** См. 8.1.2.

Тип **SHORT_NAME_Type:** См. 8.1.2.

Тип **SOURCE_DOCUMENT_Type:** См. 8.2.2.1.

Тип **SYNONYMOUS_NAME_Type:** См. 8.1.2.

Тип **TEXT_Type:** См. 8.1.2.

Тип **TRANSLATION_Type:** См. 8.1.3.

Перечень ограничительных условий:

Структура с наследованием свойств, определяемая иерархией классов (посредством XML-элемента **its_superclass,** унаследованного из комплексного XML-типа данных **CLASS_Type**), не должна содержать циклов.

Только те типы свойств, которые являются «видимыми» для класса, могут становиться применимыми для этого класса благодаря ссылке в списке, определенном с помощью совокупности элементов **described_by**.

Только те типы данных, которые являются «видимыми» для класса, могут становиться применимыми для этого класса благодаря ссылке в списке, определенном с помощью совокупности элементов **defined_types**.

Только те свойства, которые неприменимы для класса из-за наследования свойств, могут становиться применимыми для этого класса благодаря ссылке в списке, определенном с помощью совокупности элементов **described_by**.

Только те типы данных, которые неприменимы для класса из-за наследования свойств, могут становиться применимыми для этого класса благодаря ссылке в списке, определенном с помощью совокупности элементов **defined_types**.

Только контекстно-зависимые свойства (для типа **DEPENDENT_P_DET_Type,** см. раздел 6.7.4), чьи контекстные параметры (для типа **CONDITION_DET_Type,** см. 6.7.4) применимы в классе, могут становиться применимыми для этого класса благодаря ссылке с помощью его XML-элемента **described_by**.

Совокупность элементов **constraints,** должна определять ограничения, которые совместимы с областью значений свойств, для которых они применяются.

Все свойства, на которые даются ссылки в совокупности элементов **precondition** ограничительного условия и для которых основным типом данных является тип **CONFIGURATION_CONTROL_CONSTRAINT_Type,** должны быть применимы для этого класса.

Все свойства, на которые даются ссылки в совокупности элементов **constraints,** должны быть либо «видимыми», либо применимыми для этого класса.

Свойства, на которые даются ссылки в совокупности элементов **sub_class_properties,** также должны содержать ссылки и в совокупности элементов **described_by**.

Свойства, на которые даются ссылки в совокупности элементов **class_constant_values,** заявлялись как значимые для класса в некотором суперклассе текущего класса, либо в самом текущем классе.

Если свойства, на которые даются ссылки в совокупности элементов **class_constant_values,** уже были закреплены за значением в суперклассе, то закрепленное в текущем классе значение должно быть аналогичным.

Если элемент **is_deprecated** существует, то должен существовать и элемент **is_deprecated_interpretation**.

Примерные значения элемента **is_deprecated_interpretation** должны определяться в тот момент, когда принималось решение о возражении.

Если тип **PLACEMENT_TYPE_Type, AXIS1_PLACEMENT_TYPE_Type, AXIS2_PLACEMENT_2D_TYPE_Type** или **AXIS2_PLACEMENT_3D_TYPE_Type** является типом данных, присвоенным свойству, на которое дается ссылка в группе элементов **described_by,** то должен быть предоставлен соответствующий XML-элемент **geometric_representation_context**.

6.7.2.2 Класс категорий

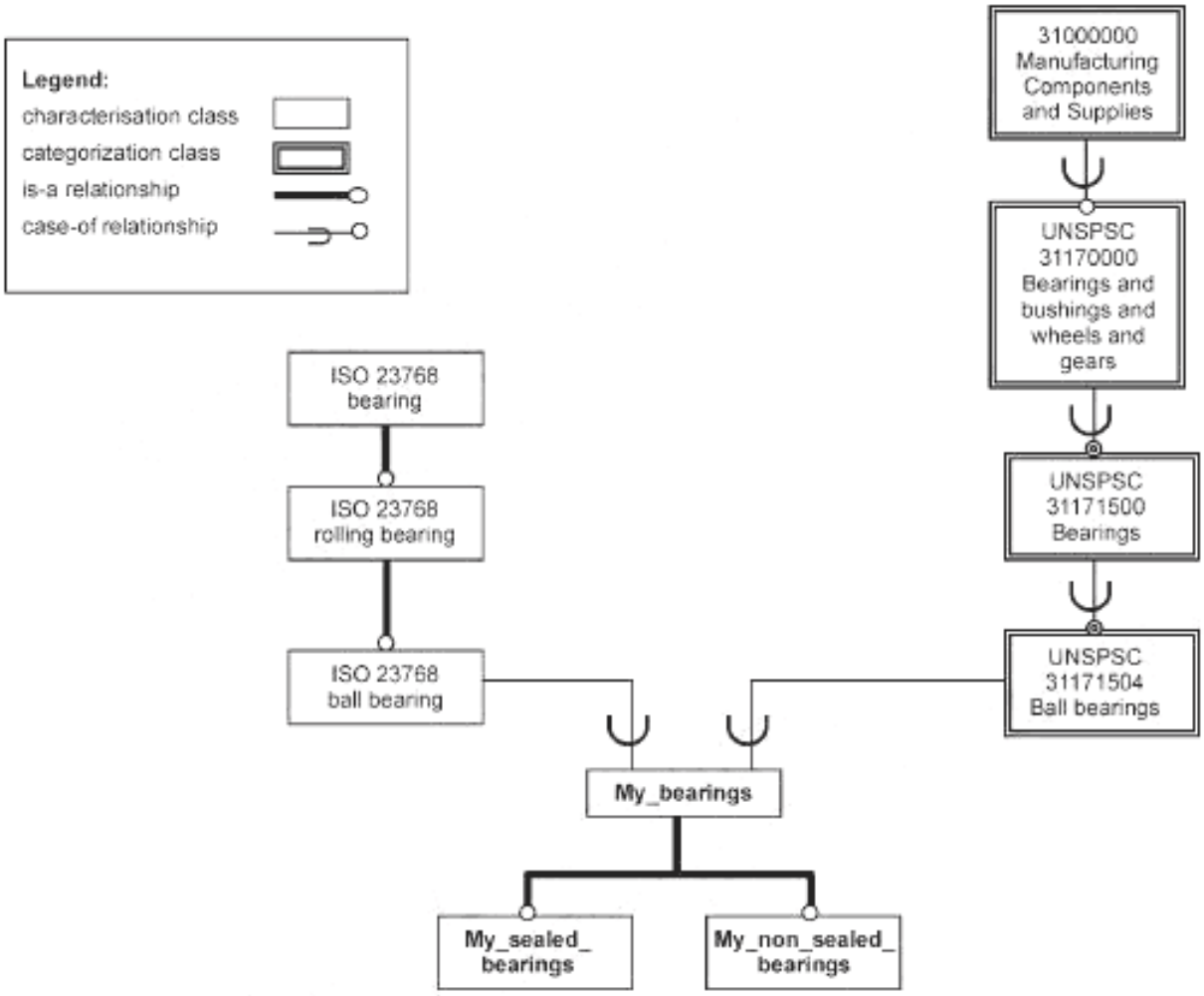
Класс категорий позволяет моделировать группирование множества объектов, содержащих элемент категоризации.

Пример 1 — Производственные компоненты и материальные запасы, Промышленная оптика — это пример класса категоризации продукции, определенной в UNSPSC.

Ни свойства, ни типы данных, ни ограничительные условия не связаны с этим классом. Более того, классы характеризации могут быть не связаны друг с другом с помощью экземплярного (is-a) соотношения наследования, однако они могут быть связаны друг с другом только с помощью условного (is-case-of) соотношения. Конкретный XML-элемент, называемый элементом **categorization_class_superclasses,** позволяет регистрировать классы категоризации, которые являются суперклассами в условной (case-of) иерархии.

Примечание — Используя условные (case-of) конструкции ресурсов, классы элементов могут также объединяться с классами категоризации.

Пример 2 — В приведенном примере показано, как классы характеризации и классы категоризации могут объединяться для достижения определенных целей. Поставщик шарикоподшипников желает разработать свою собственную онтологию и сделать ее удобной для поиска и использования. Для реализации этой задачи он/она желают использовать стандартные свойства и стандартную их классификацию. Этот поставщик предоставляет только шарикоподшипники, однако некоторые из них поставляются со смазкой, а другие — без смазки. Конкретные свойства могут связываться как со смазанными подшипниками, так и не со смазанными, однако эти категории не существуют в виде классов в стандартной онтологиях подшипников. По этой причине поставщик подшипников действует следующим образом: (1) Он/она разрабатывает собственную онтологию, состоящую из трех классов характеристик: my_bearing, my_sealed_bearings и my_non_sealed_bearing. Две последние характеристики связаны с первой характеристикой экземплярным (is-a) соотношением наследования, и все свойства, закрепленные за первой характеристикой, будут наследоваться остальными характеристиками. (2) Для использования нескольких свойств, определенных в будущем ИСО/ТС 23768-1, поставщик подшипников определяет, что его/ее класс my_bearings является условным (case-of) стандартным классом подшипников со свойством ball bearing , определенном в ИСО/ТС 23768. Посредством этого условного (case-of) соотношения он/она могут импортировать его/ее класс my_bearings стандартно-определенных свойств: bore diameter, outside diameter, ISO tolerance class. Более того, он/она создают те необходимые им свойства, которые не определены в указанном стандарте. (3) Для облегчения поиска сервера, который отображает каталог поставщика, он/она представляют небольшой фрагмент UNSPSC-классификации и условного (case-of) соотношения между UNSPSC-классом ball_bearings и своим собственным классом my_bearing. Результат выполненных операций иллюстрируется нижеприведенным рисунком 25.



Legend — Условные обозначения; Characterisation class — Класс характеризации; Categorization class — Класс категоризации; Is-a relationship — Экземплярное соотношение; Case-of relationship — Условное соотношение; ISO 23768, bearing — Стандарт ИСО 23768, Подшипник; ISO 23768 rolling bearing — Стандарт ИСО 23768, Роликовый подшипник; ISO 23768 ball bearing — Стандарт ИСО 23768, Шариковый подшипник; 31000000, Manufacturing Components and Supplies — 31000000, Производственные компоненты и материальные запасы; UNSPSC 31170000 Bearing and bushings and wheels and gears — UNSPSC 31170000, Подшипники, вкладыши, колеса и шестерни; UNSPSC 31171500, Bearings — UNSPSC 31171500 Подшипники; UNSPSC 31171504, Ball bearings — UNSPSC 31171504, Шариковые подшипники.

Рисунок 25 — Пример онтологии поставщика, использующей классы категоризации

Рисунок 26 иллюстрирует структуру класса категорий.

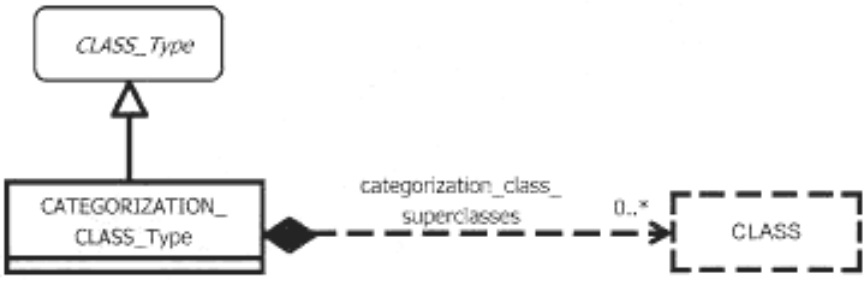


Рисунок 26 — Класс категоризации

Класс категоризации (представляемый комплексным XML-типом данных **CATEGORIZATION_CLASS_Type**) наследует XML-описание содержания, определенное в абстрактном комплексном XML-типе данных **CLASS_Type**.

Определение внутреннего элемента:

Элемент **categorization_class_superclasses:** Определяет классы категоризации, которые на один уровень выше класса категоризации в иерархии условных классов.

Перечень ограничительных условий:

Только идентификаторы, соответствующие классам, для которых основным типом данных является тип **CATEGORIZATION_CLASS_Type,** могут иметь ссылки в совокупности элементов **categorization_class_superclasses**.

Наследованный XML-элемент **its_superclass** не должен определяться при определении класса категоризации.

Наследованный XML-элемент **described_by** не должен определяться при определении класса категоризации.

Наследованный XML-элемент **defined_types** не должен определяться при определении класса категоризации.

Наследованный XML-элемент **sub_class_properties** не должен определяться при определении класса категоризации.

Наследованный XML-элемент **class_constant_values** не должен определяться при определении класса категоризации.

Наследованный XML-элемент **constraints** не должен определяться при определении класса категоризации. Класс категоризации не должен быть классом определения любого свойства.

6.7.2.3 Условный (case-of) класс элементов

Стандартные онтологии продукции разрабатываются для точного и формального представления классов продукции и ее свойств по ее определению, которое согласовано между экспертами в определенной области.

В каждой конкретной организации:

- только несколько стандартных классов характеризации продукции могут оказаться полезными;
- эти стандартные классы характеризации продукции могут быть определены в различных стандартных онтологиях продукции;
- могут существовать специфические для данной организации классы характеризации;
- только несколько стандартных свойств могут рассматриваться как необходимые, и
- некоторые специфические для данной организации свойства могут использоваться.

Таким образом, представление подобной области с непосредственным использованием стандартной онтологии продукции в пределах каждой организации не будет эффективным.

Условное (case-of) соотношение может также использоваться в стандартных словарях.

В OntoML-языке для указанной выше цели используется комплексный XML-тип **ITEM_CLASS_CASE_OF_Type** (см. рисунок 27).

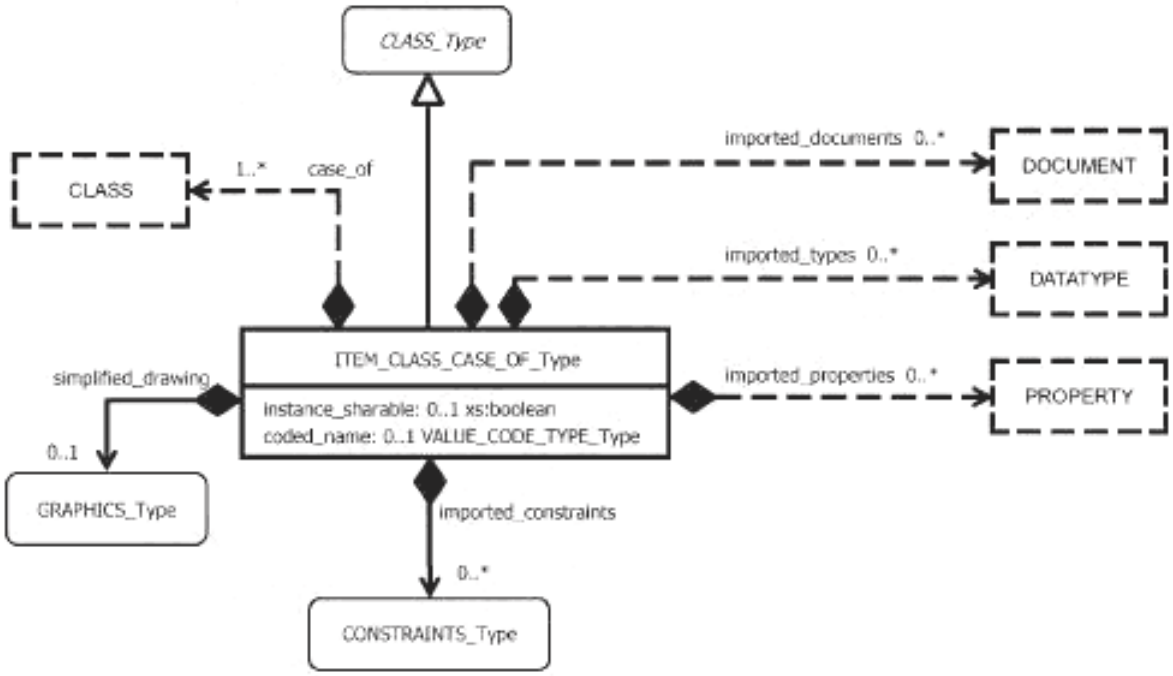


Рисунок 27 — UML-диаграмма условного класса элементов

Определения внутренних элементов:

Элемент **case_of:** Определяет класс (классы), для которого описываемый класс является условным (case-of); он представляется путем ссылки на соответствующий идентификатор (идентификаторы) понятия онтологии класса (классов).

Примечание 1 — Условное (case-of) соотношение и его использование описывается в разделе 3.5.2 Руководства ИСО/МЭК 77-2:2008.

Элемент **imported_constraints:** Определяет множество ограничительных условий, которые импортируются из класса (классов) элементов определенного условного класса элементов.

Примечание 2 — В отличие от других импортируемых объектов элемент (ограничительные условия) **imported_constraints** не может выбираться, когда разработан элемент условного класса. Эти условия ограничивают области любых свойств, определенных в совокупности элементов **imported_properties** в условном классе XML-элемента **case_of**, из которого они импортируются.

Элемент **imported_documents:** Определяет импортируемый документ (документы) из класса (классов) **case_of**, который требуется для описания текущего класса; он представляется с помощью ссылки на соответствующий идентификатор (идентификаторы) понятия онтологии класса (классов).

Элемент **imported_properties:** Определяет импортируемое свойство (свойства) из класса (классов) **case_of**, который требуется для описания текущего класса; он представляется с помощью ссылки на соответствующее свойство (свойства) понятия онтологии класса (классов).

Элемент **imported_types:** Определяет импортируемый тип (типы) из класса (классов) **case_of**, который требуется для описания текущего класса; он представляется с помощью ссылки на соответствующий тип (типы) понятия онтологии класса (классов).

Элемент **instance_sharable (ITEM_CLASS_Type):** Если этот элемент ошибочный, то он определяет, что экземпляры условного класса элементов являются характерными; если этот элемент не предоставляется или является истинным, то он определяет, что экземпляры условного класса элементов являются индивидуальными.

Примечание 3 — Общая модель словаря, определенная в ИСО 13584/МЭК 61360, является реализацией, зависящей от решения о том, должны ли представляться некоторые элементы реальных слов, моделируемые с помощью нескольких фрагментов EXPRESS-данных или того же фрагмента данных в файле обмена данными. Таким образом, экземпляр условного класса, чей элемент **instance_sharable** является неправильным и на который дается ссылка с помощью нескольких экземпляров условного класса на уровне модели данных, интерпретируется как несколько экземпляров реальных слов одного и того же характера.

Элемент **simplified_drawing:** Определяет чертеж, который может быть связан с описываемым классом.

Определение внутреннего типа:

Тип **VALUE_CODE_TYPE_Type:** Является строкой (**xs:string** XML-диаграммы), которая представляет значения, допускаемые для значений кода. Эта строка должна содержать не более 35 символов.

Определения внешних типов:

Тип **CLASS_Type:** См. 6.7.2.1.

Тип **CLASS_VALUE_ASSIGNMENT_Type:** См. 6.7.2.4.

Тип **CONSTRAINTS_Type:** См. 8.5.1.

Тип **GRAPHICS_Type:** См. 8.2.2.2.

Перечень ограничительных условий:

Структура с наследованием свойств, определяемая иерархией классов (посредством XML-элемента **its_superclass**, наследованного из комплексного XML-типа данных **CLASS_Type**), не должна содержать циклов.

Только те типы свойств, которые являются «видимыми» в классе, могут становиться применимыми для этого класса благодаря ссылке в списке, определенном с помощью совокупности элементов **described_by**.

Только те типы данных, которые являются «видимыми» в классе, могут становиться применимыми для этого класса благодаря ссылке в списке, определенном с помощью совокупности элементов **defined_types**.

Только те свойства, которые неприменимы для класса из-за наследования свойств, могут становиться применимыми для этого класса благодаря ссылке в списке, определенном с помощью совокупности элементов **described_by**.

Только те типы данных, которые неприменимы для класса из-за наследования свойств, могут становиться применимыми для этого класса благодаря ссылке в списке, определенном с помощью совокупности элементов **defined_types**.

Совокупность элементов **constaints** должна определять ограничительные условия, которые совместимы с областью значений свойств, к которым они применимы.

Все свойства, на которые даются ссылки в совокупности элементов **precondition** ограничительного условия и для которых основным типом данных является тип **CONFIGURATION_CONTROL_CONSTRAINT_Type**, должны быть применимы для этого класса.

Все свойства, на которые даются ссылки в совокупности элементов **constraints**, должны быть либо «видимыми», либо применимыми для этого класса.

Свойства, на которые даются ссылки в совокупности элементов **class_constant_values**, заявлялись как значимые для класса в некотором суперклассе текущего класса, либо в самом текущем классе.

Если свойства, на которые даются ссылки в совокупности элементов **class_constant_values**, уже были закреплены за значением в суперклассе, то закрепленное в текущем классе значение должно быть аналогичным.

Только идентификатор, соответствующий классу, для которого основным типом данных является тип **ITEM_CLASS_Type** или тип **ITEM_CLASS_CASE_OF_Type**, может давать ссылку в XML-элементе **its_superclass**.

Только идентификатор, соответствующий классу, для которого основным типом данных является тип **ITEM_CLASS_Type**, тип **ITEM_CLASS_CASE_OF_Type** или тип **CATEGORIZATION_CLASS_Type**, может давать ссылку в совокупности элементов **case_of**.

Свойства, которые приводят ссылку в совокупности элементов **sub_class_properties**, должны давать ссылку либо в совокупности элементов **described_by**, либо в совокупности элементов **imported_properties**.

Все значимые для класса свойства, заявляемые путем ссылки в группе элементов **sub_class_properties**, которые также ссылаются в группе элементов **imported_properties**, должны быть значимыми для класса свойствами во всех элементах **case_of** классов, где они применимы.

Значения, присваиваемые импортированному свойству в группе элементов **class_constant_value**, не должны отличаться от возможного значения, закрепленного за тем же свойством в ссылочном классе.

Все свойства, которые дают ссылку в элементе **imported_properties** и которые присваивают классу постоянное значение из совокупности элементов **case_of**, должны присваивать те же постоянные значения текущему классу.

Каждое ограничительное условие, определенное в группе элементов **imported_constraints** посредством XML-элемента **constraint**, должно определяться либо как ссылочное ограничительное условие с использованием XML-атрибута **constraint_ref**, либо как определенное ограничительное условие с использованием XML-элемента **constraint_definition**, либо и того и другого.

6.7.2.4 Значимое для класса свойство

Свойство может быть определено как единственное значение, принимаемое в заданном классе. Это свойство называется «значимым для класса» свойством и представляется в двух вариантах:

- как заявляемое в заданном классе; это свойство описывается как любое другое свойство, но с одним ограничением; это свойство дает ссылку в XML-элементе **sub_class_properties** связанного класса;
- как типовое значение, которое может присваиваться в нем.

Присвоение значения важного для класса свойства другому классу представляется с помощью комплексного XML-типа данных **CLASS_CONSTANT_VALUES_Type** (см. рисунок 28).

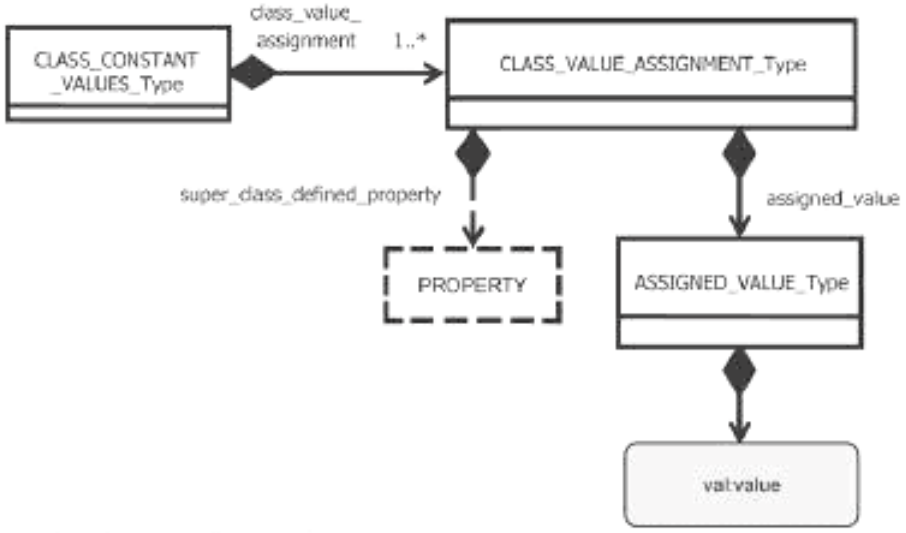


Рисунок 28 — Структура присвоенного значения класса

Определения внутренних элементов:

Элемент **class_value_assignment:** Определяет все присвоения значений в классе.

Элемент **class_value_assignment/assigned_value:** Определяет значение, присвоенное свойству и действующее для всего класса, соотнеся это присвоение в классе с совокупностью элементов **class_constant_values**.

Примечание 1 — Элемент **assigned_value** принадлежит области значений для ссылочного подкласса свойств.

Элемент **class_value_assignment/super_class_defined_property:** Определяет ссылку на свойство (определенное в суперклассе как его собственное свойство), которому присваивается значение XML-элемента **assigned_value**).

Примечание 2 — Свойство определяется как принадлежащее подклассу, когда оно появляется в XML-элементе **sub_class_properties,** определенном в комплексном XML-типе **ITEM_CLASS_Type** или **ITEM_CLASS_CASE_OF_Type**.

Определения внутренних типов:

Тип **ASSIGNED_VALUE_Type:** Является значением, которое должно присваиваиваться.

Тип **CLASS_VALUE_ASSIGNMENT_Type:** Является перечнем присвоений значений в классе.

Тип **VALUE_CODE_TYPE_Type:** Является строкой (**xs:string** XML-диаграммы), которая представляет значения, допускаемые для значений кода. Эта строка должна содержать не более 35 символов.

Определения внешних типов:

Элемент **val:value:** Является описанием типичного значения.

Примечание 3 — Элемент **val:value** определен в ИСО/ТС 29002-10 на формат обмена данными о продукции.

Перечень ограничительных условий:

Значение, присваиваемое с помощью XML-элемента **assigned_value,** должно иметь тип, совместимый с областью значений для связанного с ней типа (со ссылкой с помощью XML-элемента **super_class_defined_property**).

6.7.3 Класс онтологии повышенного уровня

В данном разделе определены компоненты моделирования, которые могут использоваться в усовершенствованном подклассе OntoML-языка.

Классы характеризации позволяют охватывать различные виды элементов в прикладной области и служат для идентификации тех характеристических свойств, которые обеспечивают дискриминационные элементы класса с помощью их значений. Идентичные значения для всех характеристических свойств означают, что элементы идентичны. Различные значения для некоторых свойств означают, что элементы различаются, поэтому характеристические свойства, как предполагается, должны быть неизменными, т. е. инвариантными для данного элемента.

Пример 1 — При взгляде на область механических фиксаторов мы можем идентифицировать класс metric threaded screw как класс характеристик фиксаторов (крепежа) и свойства типа threaded length, total length, threaded diameter, material, part number как характеристические свойства, позволяющие характеризовать различные подклассы идентичных элементов.

Свойства элемента не являются только характеристическими. В соответствии с отраслевой точкой зрения мы имеем элемент, а число других свойств считается полезным. Если пользователь онтологии обязан закупать винты (***procuments***), то будут необходимы свойства ***price*** и ***delivery delay***. Если пользователь отвечает за управление материально-техническим снабжением (***inventory management***), то необходимо свойство ***inventory size,*** т. е. число доступных винтов конкретного типа и число заказов этих винтов ***quantity of order*** и эти свойства являются наиболее значимыми. Если пользователь онтологии разрабатывает продукцию с использованием системы автоматизированного проектирования и желает ввести один винт в текущую продукцию, то свойство геометрической формы ***geometric shape*** винта сделает процесс проектирования более эффективным.

По сравнению с характеристическими свойствами рассмотренные выше свойства будут иметь два отличия, определяющие критерии их идентификации:

- каждое из этих свойств имеет смысл только с некоторых точек зрения, для некоторых областей применения или отраслей промышленности;

- большинство этих свойств не имеет характеристик элемента; они могут изменяться без изменения целевого элемента.

Для обеспечения возможности разработчика онтологии разделять эти виды свойств от характеристических свойств, а также для структурирования всех отраслевых свойств, которые могут ассоциироваться с элементом, OntoML-язык предоставляет две дополнительные категории классов:

- классы функциональных представлений, которые предоставляются для представления отраслевых точек зрения на элементы.

Пример 2 — Материально-техническое снабжение, производственные запасы, маркетинг; геометрические характеристики (трехмерные, упрощенные) или геометрическое представление (двухмерное, вид спереди, точность) являются примерами точек зрения.

Примечание 1 — Как показано в геометрическим примере, определение точки наблюдения может потребовать не только имя (***name***) ("geometry"), но и значения некоторых переменных, например ***geometry_level*** (2D, 3D), ***side*** (вид сверху, спереди...) или ***level_of_detail***. Подобные переменные называются «контрольными переменными представления (вида)»;

- классы функциональных моделей, которые предоставляются для представления элементов определенного класса (классов) характеристик в соответствии с отраслевой точкой зрения, которая определяется с помощью класса функциональных представлений.

Примечание 2 — Каждый класс функциональных моделей относится к одному классу функциональных представлений для определения отраслевой точки зрения.

Примечание 3 — Если класс функциональных моделей не соответствует классу характеристик в обмениваемой онтологии, то он будет предназначен для связи с классом характеристик на целевой системе пользователя.

Пример 3 — Класс функциональных моделей procurement может содержать значения для следующих свойств: part number (импортированного из класса характеристик metric threaded screw), price, delivery delay, для каждого винта в классе metric threaded screw.

Пример 4 — Множество двухмерных точных представлений переднего геометрического вида в классе metric: threaded screw может содержать класс функциональных моделей, которое предоставляет геометрическое представление элементов данного класса.

Примечание 4 — В контексте OntoML-языка геометрическое представление может обмениваться в виде http-файлов, чьи унифицированные идентификаторы ресурса (URI) определяются тем свойством, чьим типом данных является тип **URI_TYPE_Type**.

Стандарты комплекса ИСО 13584 не определяют, какие свойства должны представляться в классах характеризации или в классах функциональных моделей, а не в различных классах функциональных представлений, которые могут существовать. Ответственность за принятие решения относительно разбивки свойств одного и того же элемента определения классов функциональных представлений, необходимых для определения характеристик различных отраслевых точек зрения, несет разработчик онтологии.

При структурировании онтологии может приниматься во внимание ряд рекомендаций, а именно:

- свойства, представляемые в классе характеризации, должны позволять различать элементы, которые не считаются идентичными сообществом, которое будет использовать данную онтологию.

Пример 5 — Класс характеризации винта, такой, что два винта с одинаковыми значениями всех их характеристик не будут совместимыми при замене в механической продукции, возможно, не будут приспособлены для нужд пользователей-машиностроителей. Для приспособления к их нуждам необходимо создать новые свойства для разделения этих двух винтов;

- свойства, которые не являются характеристическими для элементов в классе характеризации, т. е. те значения, которые можно изменять без изменения элемента, могут рассматриваться как принадлежащие к классу функциональных моделей.

Пример 6 — Свойство price элемента может изменяться со временем без изменения самого элемента. Это свойство и возможно — некоторых других коммерчески значимых свойств, могут приниматься во внимание для создания класса функциональных моделей.

- класс функциональных моделей должен быть создан в том случае, когда пользователей заинтересуют некоторые другие свойства.

Пример 7 — Свойства, которые описывают утилизацию элемента, принадлежащего классу механических элементов, могут рассматриваться для создания класса функциональных моделей.

6.7.3.1 Класс функциональных представлений

Класс функциональных представлений позволяет характеризовать частную отраслевую точку зрения (также называемую «категорией представления»), которая может оказаться полезной для различных классов продукции.

Примечание 1 — Отраслевая точка зрения может представлять собой перечень отдельных инженерных дисциплин. В этом случае класс функциональных представлений является простым (элементарным) классом без какого-либо свойства. Имя и определение класса описывают точку зрения. Определение должно также либо указывать вид свойств, которые должны быть представлены в классах функциональных моделей и которые рассматривают данный класс функциональных представлений, либо четко перечислять свойства, которые должны представляться.

При необходимости класс функциональных представлений может содержать свойства, называемые «контрольными переменными представления», для дальнейшего определения частной субкатегории представлений.

Пример 1 — Класс функциональных представлений geometry определяет геометрическую точку зрения, однако она остается двусмысленной, для устранения которой может быть определена контрольная переменная представлений geometry_level, принимающая значение 2D или 3D. Функциональная модель, которая связывает данный класс функциональных представлений, будет импортировать эту переменную для определения того, какой вид конкретных геометрических представлений она будет предоставлять.

Пример 2 — Класс функциональных представлений, называемый acceptable_environmental_condition, может использоваться для определения характеристик, при которых экологическая продукция может безопасно использоваться. Определение указанного класса может, например, указывать в своем определении, что классы функциональных моделей, связанных с данным классом функциональных представлений, могут содержать только (1) свойства, импортируемые из класса элементов и (2) экологические свойства значимого для типа уровня, имеющего по крайней мере два значения — min и max, для определения того, в какой контекстной продукции ссылочного класса характеристик будет гарантироваться применение. Подобный класс функциональных представлений может определяться без использования контрольных переменных представления.

Пример 3 — Класс функциональных представлений может также использоваться для описания точки зрения inventory status. Указанный класс может определяться без какой-либо контрольной переменной представления, если разработчик онтологии желает только определять уникальную точку зрения inventory status, которая будет описывать для каждого вида продукции свойство inventory size, т. е. число каждой детали, которая в настоящее время имеется в наличии, и свойство quantity of order детали.

Контрольная переменная представления не характеризует элемент, а дает лишь его представление, поэтому она должна определяться свойством типа **REPRESENTATION_P_DET**, определенном в 6.7.5.

В настоящем стандарте класс функциональных представлений устанавливается с помощью комплексного XML-типа данных **NON_INSTANCIABLE_FUNCTIONAL_VIEW_CLASS_Type** (см. рисунок 29).

Примечание 2 — Имя "non instanciable functional view class type (тип не представляемого класса функциональных представлений)" заимствовано из ИСО 13584-24:2003, в котором также определяется метод для программного представления созданного элемента, поэтому в данном случае можно говорить, что представления (виды) являются характеристическими. Этот метод, редко используемый в прошлом, не вводится в OntoML-язык, однако имя "non instanciable functional view class type (тип характерного класса функциональных представлений)" будет сохраняться в одних и тех же ресурсах ИСО 13584-24:2003 и OntoML-языка.

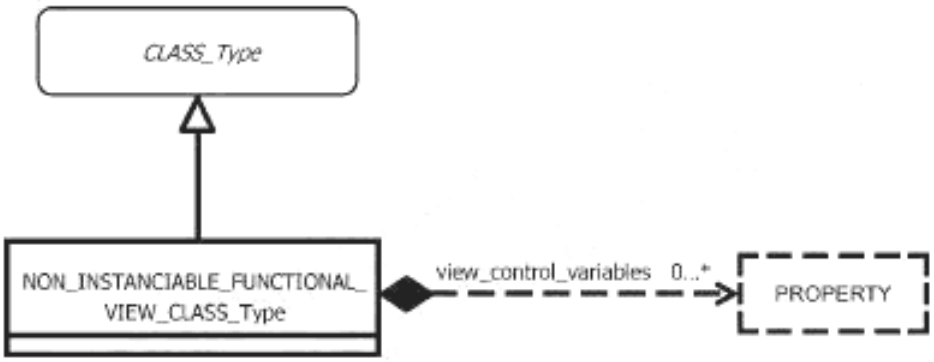


Рисунок 29 — UML-диаграмма понятия класса в онтологии повышенного уровня

Определение внутреннего элемента:

Элемент **view_control_variables:** Определяет список свойств, которые в дальнейшем будут определять отраслевую точку зрения, определяемую с помощью класса функциональных представлений.

Определение внешнего типа:

Тип **CLASS_Type:** См. 6.7.2.1.

Перечень ограничительных условий:

Либо данный класс не имеет ссылочного суперкласса (наследованного XML-элемента **its_superclass**), либо, если имеется один класс, то ссылочным суперклассом должен быть тип данных **NON_INSTANCIABLE_FUNCTIONAL_VIEW_CLASS_Type**.

Типом каждой контрольной переменной представления (предметный XML-элемент комплексного XML-типа данных **PROPERTY_Type**) должен быть тип **QUANTITATIVE_INT_TYPE_Type** (см. 8.3.8), чьими значениями являются последовательные целые числа.

Свойства, предназначенные для использования в качестве контрольных переменных представления, должны идентифицироваться как тип данных **REPRESENTATION_P_DET_Type** (см. 6.7.5) и должны давать ссылку в группу элементов **described_by** нереализуемого класса функциональных представлений.

6.7.3.2 Класс функциональных моделей

Класс функциональных моделей предназначен для описаний с отраслевой точки зрения, определенных с помощью класса функциональных представлений элементов, принадлежащих какому-либо классу характеристик элементов.

Пример 1 — Предположим, что класс screw class является корневым классом элементов в иерархии классов винтов, а также то, что этот класс заявляется как обладающий свойством part number, позволяющим идентифицировать винт любого субкласса screw class. Предположим также, что item price view является классом функциональных представлений, чьим определением является «классы функциональных моделей, которые соответствуют этому представлению и должны предоставлять цены в евро»; после этого класс функциональных моделей screw price model может быть определен и будет относиться и к классу screw (с помощью XML-элемента view_of, см. рисунок 31) — для определения того, что предоставляет функциональные модели винтов, а также и к классу item price view (с помощью XML-элемента created_view, см. рисунки 30 и 31) — для определения того, что он предоставляет свойство item price view view. Данный класс функциональных моделей может также импортировать свойство part number из класса screw (посредством XML-элемента imported_properties_from_item, см. рисунок 31) и заявлять (с помощью XML-элемента described_by) свойство euro price как свойство real_currency_type. При этих предположениях каждый экземпляр класса screw price model может содержать пару свойств (part number, euro price) , которая будет определять цену одного винта в классе screw (или в любом из его субклассов) в евро.

Примечание 1 — В примере 1 цены могут рассчитываться автоматически для каждого винта из класса *screw* путем определения того, что свойство ***part number*** должно представляться как в классе винтов, так и в классе функциональных моделей, а также может использоваться для расчета внешней связи между содержаниями обоих классов (см. XML-элемент **required_item_values** в 7.4).

Пример 2 — При том же определении screw class и item price view, что и в примере 1, конкретный класс функциональных моделей может быть определен для каждого из субклассов класса screw class, содержащих экземпляры этого класса.

Примечание 2 — В этом случае классы функциональных моделей *априори* связываются с классом характеристик продукции и в данном случае они представляются с помощью субтипа типа данных **FM_CLASS_VIEW_OF_Type**, однако класс функциональных моделей не требует относиться к какому-нибудь классу характеризации продукции. Это позволяет *апостериорно* связывать их с существующими классами характеризации.

Примечание 3 — Указанная апостериорная связь определяется с помощью комплексного XML-типа данных **A_POSTERIORI_VIEW_OF_Type**, определенного в 8.6.2.

Класс функциональных моделей может импортировать:

- свойства и/или типы и/или документы из класса функциональных представлений, который определяет точку зрения предоставляемых ориентированных описаний;
- свойства и/или типы и/или документы из класса (классов) функциональных моделей, для которого текущий класс функциональных моделей может быть условным (case-of).

Примечание 4 — Класс функциональных моделей может также наследовать свойства и/или типы данных и/или документы из возможного суперкласса, что позволяет совместно использовать одни и те же свойства и/или типы данных и/или документы между иерархиями классов функциональных моделей.

Подобный класс функциональных моделей, представляемый с помощью комплексного XML-типа данных **FUNCTIONAL_MODEL_CLASS_Type**, показан на рисунке 30.

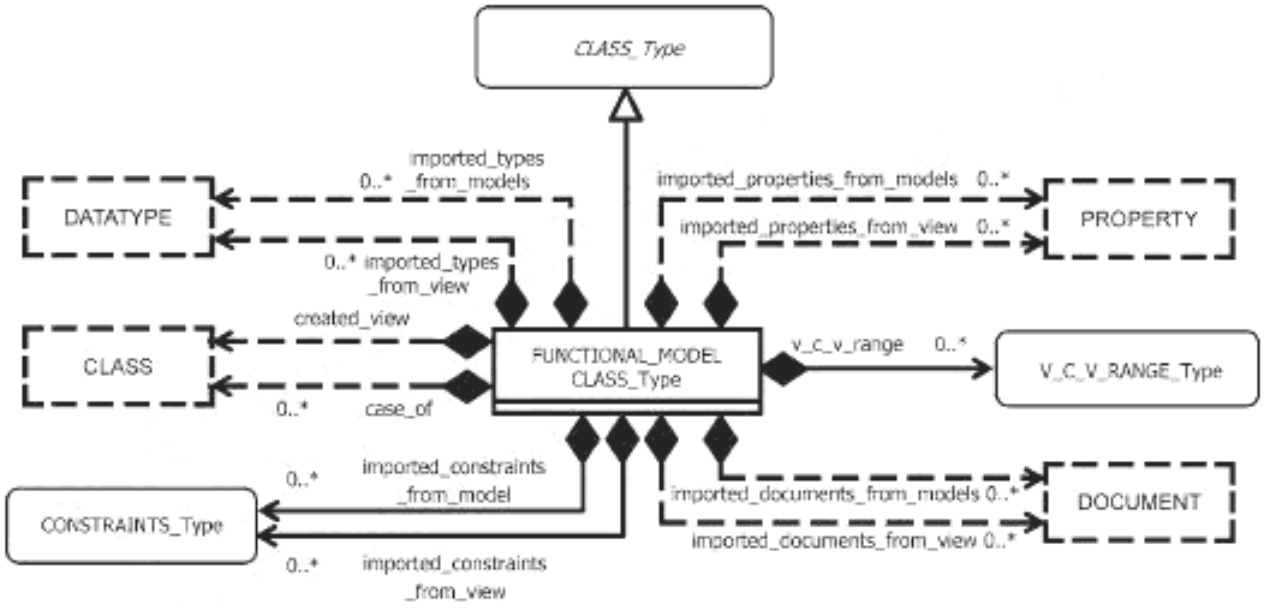


Рисунок 30 — UML-диаграмма онтологического понятия повышенного класса

Определения внутренних элементов:

Элемент **case_of:** Определяет возможные классы функциональных моделей, если текущий класс функциональной модели является условным (*case-of*).

Примечание 5 — Условное соотношение и его использование описано в разделе 3.5.2 ИСО/МЭК 77-2:2008.

Элемент **created_view:** Определяет класс функциональных представлений, который характеризует точку зрения, рассматриваемую классом функциональных моделей.

Элемент **imported_constraints_from_model:** Определяет возможные ограничительные условия, которые импортированы из условных классов функциональных моделей.

Примечание 6 —Все ограничительные условия, которые применимы к свойствам модели и импортированные классом функциональных моделей, импортируются в данный класс.

Элемент **imported_constraints_from_view:** Определяет возможные ограничительные условия, которые импортированы из созданного представления.

Примечание 7 — Все ограничительные условия, которые применимы к свойствам представления и импортированные классом функциональных моделей, импортируются в данный класс.

Элемент **imported_documents_from_model:** Определяет возможный документ, который импортирован из условных классов функциональных моделей.

Элемент **imported_documents_from_view:** Определяет возможный документ, который импортирован из созданного представления.

Элемент **imported_properties_from_model:** Определяет возможные свойства, которые импортированы из условных классов функциональных моделей.

Элемент **imported_properties_from_view:** Определяет возможные свойства, которые импортированы из созданного представления.

Элемент **imported_types_from_model:** Определяет возможные типы данных, которые импортированы из условных классов функциональных моделей.

Элемент **imported_types_from_view:** Определяет возможные типы данных, которые импортированы из созданного представления.

Элемент **v_c_v_range:** Определяет список диапазонов контрольных переменных представлений, определяющих различные отраслевые субкатегории класса функциональных моделей. Каждая из них должна соответствовать контрольной переменной функционального представления, на которую дается ссылка с помощью XML-элемента **created_view**. Когда контрольная переменная функционального представления, определенная с помощью XML-элемента **created_view**, не представлена в элементе **v_c_v_range**, то ее диапазоном будет полная область значений.

Примечание 8 — В большинстве случаев класс функциональных представлений не будет иметь контрольной переменной представления, поэтому элемент **v_c_v_range** каждого класса функциональных моделей, который соответствует данному классу функциональных представлений, будет пустым.

Определения внешних типов:

Тип **CLASS_Type:** См. 6.7.2.1.

Тип **CONSTRAINTS_Type:** См. 8.5.1.

Тип **V_C_V_RANGE_Type:** См. 6.7.3.4.

Перечень ограничительных условий:

Класс, на который дается ссылка с помощью XML-элемента **created_view,** должен иметь в качестве основного типа данных **NON_INSTANCIABLE_FUNCTIONAL_VIEW_CLASS_Type**.

Каждая контрольная переменная представления, используемая в XML-элементе **v_c_v_range,** должна соответствовать контрольной переменной функционального представления, на которую дается ссылка с помощью элемента **created_view**.

Каждая контрольная переменная представления, определенная в ссылочном представлении (XML-элемент **created_view**) и используемая в XML-элементе **v_c_v_range,** должна давать ссылку в группу элементов **imported_properties_from_view**.

Примечание 9 — Каждая контрольная переменная представления, чей XML-элемент **v_c_v_range** не ограничивается одноточечным множеством, является частью кода класса функциональных моделей. Последнее определено в ограничительных условиях для типа данных **EXPLICIT_FUNCTIONAL_MODEL_CLASS_EXTENSION_Type** (см. раздел 7.4).

Либо класс не имеет суперкласса, либо ссылочный суперкласс (наследованный XML-элемент **its_superclass**) должен иметь в качестве основного тип данных **FUNCTIONAL_MODEL_CLASS_Type** или **FM_CLASS_VIEW_OF_Type**.

Группа элементов **v_c_v_range** должна содержать уникальный элемент **view_control_variable_range** для каждого ссылочного свойства (XML-элемент **parameter_type**).

Каждое свойство, которое определено (с помощью XML-элемента **described_by**) или наследовано для класса функциональных моделей, может иметь либо тип данных **NON_DEPENDENT_P_DET_Type, DEPENDENT_P_DET_Type, CONDITION_DET_Type** либо **REPRESENTATION_P_DET_Type**.

Примечание 10 — Данное ограничительное условие является более ограничивающим, чем приведенное в ИСО 13584-24:2003, в котором все свойства в классе функциональных моделей принадлежат классу **REPRESENTATION_P_DET_Type**.

Каждый класс, который ссылается посредством XML-элемента **case_of,** должен принадлежать классу **FUNCTIONAL_MODEL_CLASS_Type** или **FM_CLASS_VIEW_OF_Type**.

Все значимые для класса свойства, заявляемые как ссылочные в группе элементов **sub_class_properties**, и на которые есть ссылка в группе элементов **imported_properties_from_model,** должны быть значимыми для всех классов **case_of,** к которым они применимы.

Значения, присваиваемые импортированному свойству в группе элементов **class_constant_values,** не должны отличаться от возможных значений, присваемых тому же свойству в ссылочных классах.

Каждое ограничительное условие, определенное в группах элементов **imported_constraints_from_model** и **imported_constraints_from_view** посредством XML-элемента **constraint,** должно определяться либо как ссылочное ограничительное условие (с помощью XML-атрибута **constraint_ref**), либо как заданное ограничительное условие с использованием XML-элемента **constraint_definition,** либо и того и другого.

6.7.3.3 Производный класс функциональных моделей

Производный (*view-of*) класс функциональных моделей — это класс, чьи описания с различных точек зрения непосредственно связываются с продукцией, принадлежащей особому классу характеристик продукции. В этом случае и в дополнении к свойствам и/или типам и/или документам (возможно, импортированным из ссылочного функционального представления или из условной функциональной модели (моделей), свойств и/или типов и/или документов) они могут импортироваться из класса определения характеристик продукции, для которого существующий класс функциональных моделей определен.

Примечание 1 — В частности, рекомендуется, чтобы некоторые свойства, применимые к классу характеристик элементов, импортировались для соединения каждого представления (определенного с помощью производного класса функциональных моделей) с каждым элементом класса элементов.

Подобный класс функциональных моделей представляется с помощью комплексного XML-типа данных **FM_CLASS_VIEW_OF_Type** (см. рисунок 31).

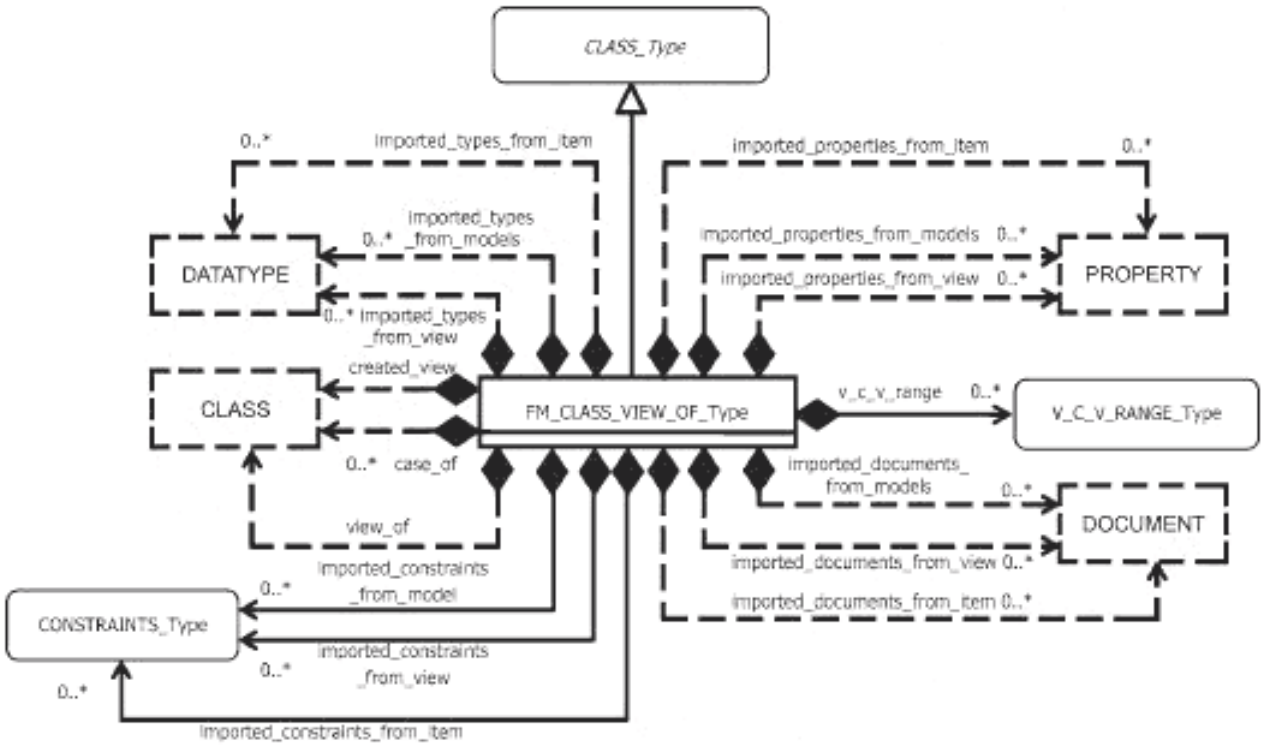


Рисунок 31 — UML-диаграмма онтологического понятия повышенного класса: Производный класс функциональных моделей

Определения внутренних элементов:

Элемент **case_of:** Определяет другие возможные классы функциональных моделей, текущим классом функциональных моделей которых является условный (*case-of*) класс.

Примечание 2 — Условное (case-of) соотношение и его использование описываются в разделе 3.5.2 ИСО/МЭК 77-2:2008.

Элемент **created_view:** Определяет класс функциональных представлений, который характеризует точку зрения пользователя, определяемую классом функциональных моделей.

Элемент **view_of:** Определяет класс определения характеристик продукции, для которого описываемый класс функциональных возможностей способен определить представления.

Элемент **imported_constraints_from_item:** Определяет возможные ограничительные условия, которые импортируются из класса элементов, для которого существующий класс способен создавать представление.

Примечание 3 — Все ограничительные условия, которые применимы к свойствам элемента и полностью импортированы с помощью класса функциональных моделей, импортируются в данный класс.

Элемент **imported_constraints_from_model:** Определяет возможные ограничительные условия, которые импортируются из условных (*case-of*) классов функциональных моделей.

Примечание 4 — Все ограничительные условия, которые применимы к свойствам модели, импортируются в данный класс.

Элемент **imported_constraints_from_view:** Определяет возможные ограничительные условия, которые импортируются из созданного представления.

Примечание 5 — Все ограничительные условия, которые применимы к свойствам представления, импортируются с помощью класса функциональных моделей в данный класс.

Элемент **imported_properties_from_item:** Определяет возможные свойства, которые импортируются из класса элементов, для которого существующий класс способен создавать представление.

Элемент **imported_properties_from_model:** Определяет возможные свойства, которые импортируются из условных классов функциональных моделей.

Элемент **imported_properties_from_view:** Определяет возможные свойства, которые импортируются из созданного представления.

Элемент **imported_types_from_item:** Определяет возможные типы данных, которые импортируются из класса элементов, для которого существующий класс способен создавать представление.

Элемент **imported_types_from_model:** Определяет возможные типы данных, которые импортируются из условных классов функциональных моделей.

Элемент **imported_types_from_view:** Определяет возможные типы данных, которые импортируются из созданного представления.

Элемент **imported_documents_from_item:** Определяет возможные документы, которые импортируются из класса элементов, для которого существующий класс способен создавать представление.

Элемент **imported_documents_from_model:** Определяет возможные документы, которые импортируются из условных классов функциональных моделей.

Элемент **imported_documents_from_view:** Определяет возможные документы, которые импортируются из созданного представления.

Элемент **v_c_v_range:** Определяет список диапазонов контрольных переменных представлений, определяющий отраслевые субкатегории класса функциональных моделей. Каждая из них должна соответствовать контрольной переменной функционального представления, на которую дается ссылка с помощью XML-элемента **created_view**. Когда контрольная переменная функционального представления, определенная с помощью XML-элемента **created_view,** не представлена в элементе **v_c_v_range,** то ее диапазоном будет полная область значений.

Примечание 6 — В большинстве случаев класс функциональных представлений не будет иметь контрольной переменной представления, поэтому элемент **v_c_v_range** каждого класса функциональных моделей, который соответствует данному классу функциональных представлений, будет пустым.

Определения внешних типов:

Тип **CLASS_Type:** См. раздел 6.7.2.1.

Тип **CONSTRAINTS_Type:** См. 8.5.1.

Тип **V_C_V_RANGE_Type:** См. 6.7.3.4.

Перечень ограничительных условий:

Класс, на который дается ссылка с помощью XML-элемента **created_view,** должен иметь в качестве основного тип данных **NON_INSTANCIABLE_FUNCTIONAL_VIEW_CLASS_Type**.

Каждая контрольная переменная представления, используемая в XML-элементе **v_c_v_range,** должна соответствовать контрольной переменной функционального представления, на которую дается ссылка с помощью XML-элемента **created_view**.

Каждая контрольная переменная представления, определенная в ссылочном представлении (с помощью XML-элемента **created_view**) и используемая в XML-элементе **v_c_v_range,** должна иметь ссылку в группе элементов **imported_properties_from_view**.

Примечание 7 — Каждая контрольная переменная представления, чей XML-элемент **v_c_v_range** не ограничивается одноточечным множеством, является частью кода класса функциональных моделей. Последнее определено в ограничительных условиях для типа данных **EXPLICIT_FUNCTIONAL_MODEL_CLASS_EXTENSION_Type** (см. 7.4).

Либо класс не имеет суперкласса, либо ссылочный суперкласс (наследованный XML-элемент **its_superclass**) должен иметь в качестве основного тип данных **FUNCTIONAL_MODEL_CLASS_Type** или **FM_CLASS_VIEW_OF_Type**.

Группа элементов **v_c_v_range** должна содержать уникальный элемент **view_control_variable_range** для каждого ссылочного свойства (XML-элемент **parameter_type**).

Каждое свойство, которое определено (с помощью XML-элемента **described_by**) или унаследовано для класса функциональных моделей, может иметь либо тип данных **NON_DEPENDENT_P_DET_Type, DEPENDENT_P_DET_Type, CONDITION_DET_Type** либо **REPRESENTATION_P_DET_Type**.

Примечание 8 — Данное ограничительное условие является менее ограничивающим, чем условие, определенное в ИСО 13584-24:2003, в котором ограничиваются все свойства класса функциональных моделей свойствами типа данных **REPRESENTATION_P_DET_Type**.

Каждый класс, на который дается ссылка посредством XML-элемента **case_of**, должен быть типа **FUNCTIONAL_MODEL_CLASS_Type** или **FM_CLASS_VIEW_OF_Type**.

Все значимые для класса свойства, заявляемые как ссылочные в группе элементов **sub_class_properties**, которые также имеют ссылку в группе элементов **imported_properties_from_model**, должны быть значимыми свойствами во всех элементах **case_of классов**, где они применимы.

Значения, присваиваемые импортируемому свойству в группе значений **class_constant_values**, не должны отличаться от возможного значения, присвоенного такому же свойству в ссылочных классах.

Каждое ограничительное условие, определенное в группах элементов **imported_constraints_from_model** и **imported_constraints_from_view** посредством XML-элемента **constraint**, должно определяться либо как ссылочное ограничительное условие (с помощью XML-атрибута **constraint_ref**), либо как конкретное ограничительное условие (с помощью XML-элемента **constraint_definition**), но не с помощью обоих элементов.

6.7.3.4 Диапазон контрольных переменных представления

Функциональные представления могут дополнительно определяться с помощью контрольных переменных представления, связанных с функциональной моделью, которая ссылается на функциональное представление, а каждый диапазон контрольных переменных представления определяет, какие конкретные представления описываются с помощью этой модели.

Пример — Предположим, что функциональное представление geometry определяет контрольную переменную представления detail_level, чьей заданной областью значений является область {simplified, standard, extended}. Частная функциональная модель может описывать только представления simplified и standard. В этом случае диапазоном контрольных переменных будет [simplified: standard].

Примечание — Если контрольная переменная функционального представления, ссылка на которую приводится в функциональной модели, не представляется с помощью диапазона значений данной контрольной переменной, то этим диапазоном будет ее полная область значений.

Диапазон контрольных переменных представления определяется с помощью комплексного XML-типа данных **VIEW_CONTROL_VARIABLE_RANGE_Type**, причем все эти диапазоны объединяются в хранилище определенное с помощью типа **V_C_V_RANGE_Type** (см. рисунок 32).

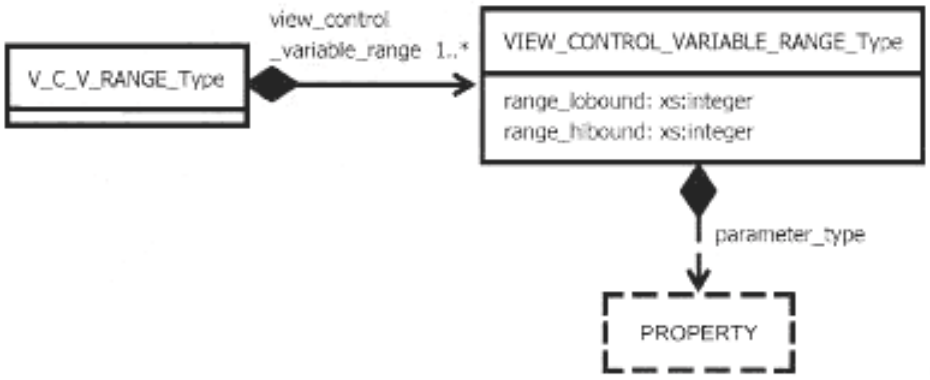


Рисунок 32 — Структура контрольной переменной представления

Определения внутренних элементов:

Элемент **view_control_variable_range:** Определяет диапазоны контрольной переменной представления для функциональной модели.

Элемент **view_control_variable_range/parameter_type:** Определяет ссылку на свойство, которое является контрольной переменной представления, для которого применим диапазон этой переменной.

Элемент **view_control_variable_range/range_hibound:** Определяет целое число, которое характеризует верхнюю границу определенного диапазона.

Элемент **view_control_variable_range/range_lobound:** Определяет целое число, которое характеризует нижнюю границу определенного диапазона.

Определение внутреннего типа:

Тип **VIEW_CONTROL_VARIABLE_RANGE_Type:** Является описанием диапазона контрольных переменных представлений.

Перечень ограничительных условий:

Свойство, на которое ссылка дается с помощью XML-элемента **parameter_type**, должно иметь тип данных **NON_QUANTITATIVE_INT_TYPE_Type_type**.

Значение XML-элемента **range_lobound** должно быть меньше или равно значению XML-элемента **range_hibound**.

Элементы **range_lobound** и **range_hibound** должны принадлежать области значений ссылочного свойства (XML-элемент **parameter_type**).

6.7.4 Онтологическое свойство простого уровня

В CIIM-модели значения свойства являются либо простыми (тип целых чисел или строк), либо другими элементами класса. Кроме того, CIIM-модель различает:

- свойства, которые могут использоваться для описания характеристик элемента, и
- свойства, которые могут использоваться только в классах функциональных моделей или функциональных представлений.

В данном разделе определены отнологические свойства простого (элементарного) уровня, которые являются свойствами, используемыми для описания характеристик элементов.

Наиболее широко используемыми в любой онтологии свойствами являются характеристические свойства, которые связаны с элементом либо с помощью значений, либо с помощью других элементов. Подобное свойство представляется с помощью комплексного XML-типа данных **NON_DEPENDENT_P_DET_Type** (см. рисунок 33).

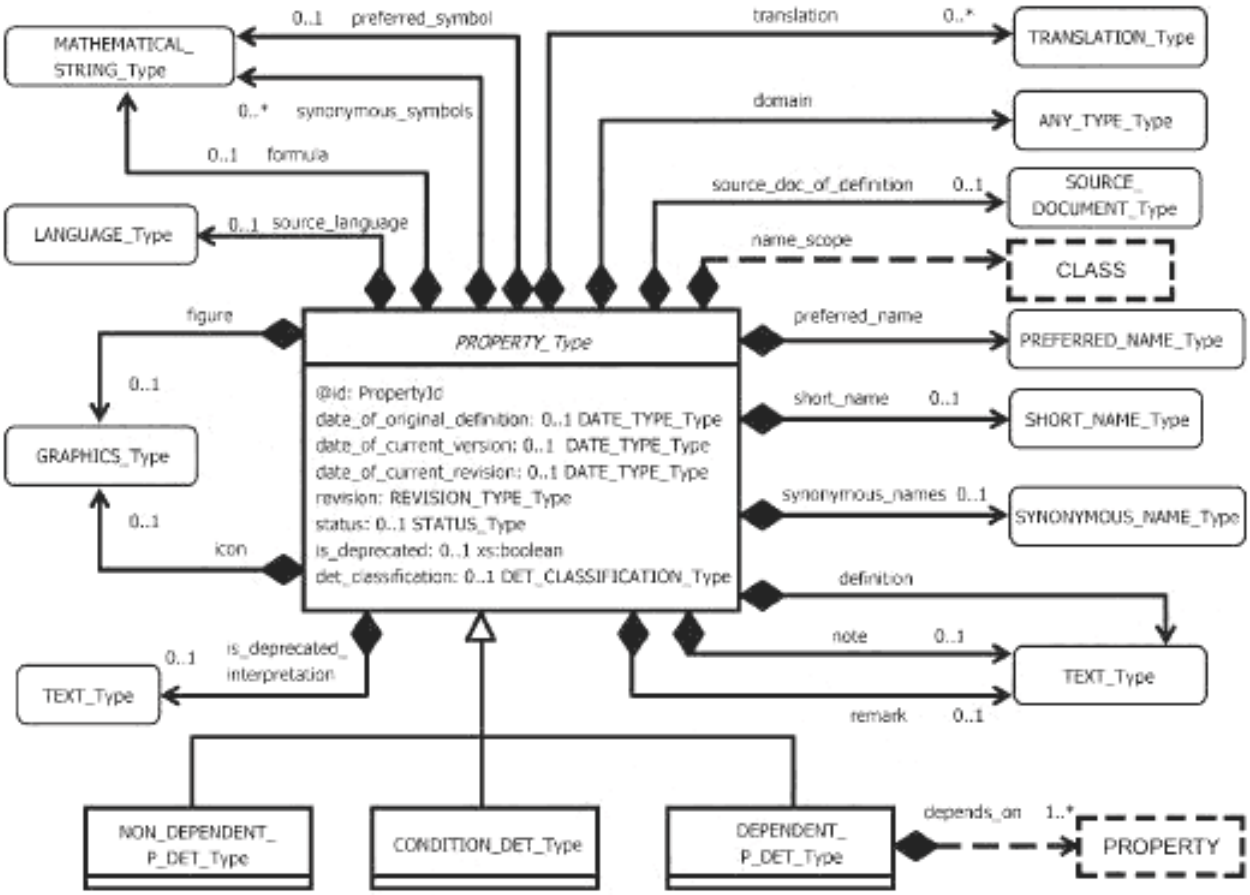


Рисунок 33 — UML-диаграмма понятия простого онтологического свойства

Примечание 1 — Характеристическое свойство определено в разделе 4 ИСО/МЭК 77-2:2008.

Однако в реальном мире никакой объект не может считаться изолированным от окружающей среды. Количественные свойства, которые могут измеряться, должны связываться с условиями, при которых они были получены.

Примечание 2 — Любой экземпляр класса размеров может иметь ссылку на температуру, при которой производилось измерение, однако на практике либо в определении свойства должна указываться ситуация при измерениях, либо она может считаться несущественной. Тем не менее, например, свойство ***resistance*** для элемента ***electric thermistor*** может сильно зависеть от свойства ***ambient temperature,*** поэтому рекомендуется всегда предоставлять подобную информацию.

Таким образом, в дополнении к обычным характеристическим свойствам, которые могут рассматриваться как контекстно-независимые, можно определять контекстные параметры посредством комплексного XML-типа данных **CONDITION_DET_Type,** а также контекстно-зависимые свойства — посредством комплексного XML-типа данных **DEPENDENT_P_DET_Type**. Для последнего вида свойств контекст (XML-элемент **depends_on**) определяется путем ссылки на глобальный идентификатор (идентификаторы) контекстного параметра (параметров).

Примечание 3 — Свойства и контекст оценок задокументирован в разделе 4.4 ИСО/МЭК 77-2:2008 для описания свойств и классов продукции.

Примечание 4 — Основное представление подобного свойства требует только определения идентификатора номера редакции **revision,** элемента **preferred_name,** определения **definition** и области его значений **domain**.

Определения внутренних элементов:

Элемент **@id:** Определяет идентификатор свойства.

Элемент **date_of_current_revision:** Определяет дату, связанную с существующей редакцией определения свойства.

Элемент **date_of_current_version:** Определяет дату, связанную с существующей версией определения свойства.

Элемент **date_of_original_definition:** Определяет дату, связанную с первой неизменной версией определения свойства.

Элемент **definition:** Определяет текст, описывающий данное свойство (возможно, переведенный).

Элемент **depends_on** (типа **DEPENDENT_P_DET_type**): Определяет множество ссылок, идентифицирующих свойства, от которых они зависят.

Элемент **det_classification:** Определяет код, представляющий ИСО 80000 (ранее — ИСО 31) для класса данного свойства.

Примечание 5 — В ИСО 13584-42:2010 определены коды, используемые в ИСО 80000 (ранее — ИСО 31), для классов, к которым может относиться количественное или неколичественное свойство.

Элемент **domain:** Определяет тип данных (область значений), связанный с данным свойством.

Элемент **figure:** Определяет возможные графические материалы, которые описывают данное свойство.

Элемент **formula:** Определяет математическое выражение, поясняющее данное свойство.

Элемент **icon:** Определяет пиктограммы, иллюстрирующие описание и связанные с именами.

Элемент **is_deprecated:** Определяет булев элемент, который указывает (если он истинен), что данное определение свойства не должно больше использоваться.

Элемент **is_deprecated_interpretation:** Определяет способ интерпретации причины исключения, примерные значения от поставщика и соответствующий идентификатор.

Элемент **name_scope:** Определяет область класса данного свойства.

Элемент **note:** Определяет дополнительную информацию относительно любой части свойства, которая существенна для его понимания (возможно, переведенную).

Элемент **preferred_name:** Определяет имя свойства, которое является предпочтительным для его использования (возможно, переведенное).

Элемент **preferred_symbol:** Определяет укороченное описание данного свойства.

Элемент **remark:** Определяет пояснительный текст, дополнительно разъясняющий содержание данного свойства (возможно, переведенный).

Элемент **revision:** Определяет номер версии существующего определения свойства.

Элемент **short_name:** Определяет сокращение предпочтительного имени свойства (возможно, переведенное).

Элемент **source_doc_of_definition:** Определяет возможный документ-источник, из которого данное определение было заимствовано.

Элемент **source_language:** Определяет язык, на котором первоначально было дано описание класса и на котором будет сохраняться исходное содержание в случае несоответствия перевода.

Элемент **status:** Определяет состояние в жизненном цикле определения класса.

Примечание 6 — Допускаемые коды состояния определяются путем заключения частного соглашения между поставщиком словаря и его пользователями.

Примечание 7 — Если XML-элемент состояния не предоставляется и если данное определение свойства не исключается как обозначенное с помощью возможного XML-элемента **is_deprecated**, то это определение свойства будет иметь то же состояние стандартизации, что и в целом у используемой онтологии. В частности, если онтология стандартизирована, то это свойство будет являться частью текущей редакции стандарта.

Элемент **synonymous_names:** Определяет набор синонимических имен предпочтительного имени свойства (возможно, переведенных).

Элемент **synonymous_symbols:** Определяет набор синонимических имен предпочтительных символов свойства.

Элемент **translation:** Определяет возможный массив переведенной информации, предоставляемый для переводимых элементов.

Определения внутренних типов:

Тип **DATE_TYPE_Type:** Является идентификатором значений, допустимых для обозначения даты (**xs:date** XML-схемы типов данных).

Тип **REVISION_TYPE_Type:** Является строкой (**xs:string** XML-схемы типов данных), представляющей значения, допускаемые для обозначения редакции. Эта строка должна содержать не более трех символов.

Тип **STATUS_Type:** Является строкой (**xs:string** XML-схемы типов данных), представляющей значения, допускаемые для обозначения состояния.

Определения внешних типов:

Элемент **PropertyId:** См. 9.1.

Тип **ANY_TYPE_Type:** См. 8.3.

Тип **GRAPHICS_Type:** См. 8.2.2.2.

Тип **LANGUAGE_Type:** См. 8.1.1.

Тип **MATHEMATICAL_STRING:** См. 8.8.2

Тип **PREFERRED_NAME_Type:** См. 8.1.2.

Тип **SHORT_NAME_Type:** См. 8.1.2.

Тип **SOURCE_DOCUMENT_Type:** См. 8.2.2.1.

Тип **SYNONYMOUS_NAME_Type:** См. 8.1.2.

Тип **TEXT_Type:** См. 8.1.2.

Тип **TRANSLATION_Type:** См. 8.1.3.

Перечень ограничительных условий:

Группа элементов **depends_on** должна относиться к свойствам, для которых основным типом данных является тип **CONDITION_DET_Type**.

Группа элементов **depends_on** не должна содержать дублированных ссылок на свойство. Если элемент **is_deprecated** существует, то также должен существовать и элемент **is_deprecated_interpretation**.

Значения экземпляра класса элемента **is_deprecated_interpretation** должны быть определены во время принятия решения об его исключении.

6.7.5 Онтологическое свойство повышенного уровня

В классах онтологических свойств повышенного уровня, т. е. в классах функциональных представлений (см. 6.7.3.1) и классах функциональных моделей (см. 6.7.3.2 и 6.7.3.3) некоторые свойства используются не для описания элементов, а для указания характеристик представления элемента. Это в особенности относится к контрольным переменным представления, которые должны представляться как тип данных **REPRESENTATION_P_DET_Type**. Более точно эти контрольные переменные должны представляться как тип данных **REPRESENTATION_P_DET_Type**. Другие свойства, определенные в классах онтологических свойств повышенного уровня, могут представляться либо как онтологическое свойство простого уровня (если они считаются описывающими некоторые аспекты элемента), либо как тип данных **REPRESENTATION_P_DET_Type** (если они характеризуют некоторое представление элемента). Тип данных **REPRESENTATION_P_DET_Type** иллюстрируется рисунком 34.

Пример — Предположим, что существует свойство P1, определенное в классе функциональных моделей, которое характеризует различные двухмерные чертежи элементов класса элементов, а также свойство P2, определенное в том же классе функциональных моделей, которое характеризует проекцию на каждом чертеже (вид сверху, снизу, спереди, сзади ...), представляемую свойством P1. Значение для свойства P1 определяет определенный аспект элемента, а значение для свойства P2 не определяет аспекты элементов. Свойство P2 должно представляться как тип данных REPRESENTATION_P_DET_Type, а свойство может представляться как элемент non_dependent_P_DET.

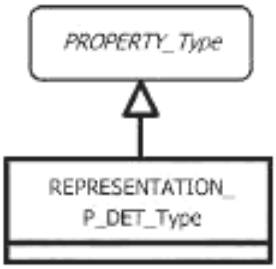


Рисунок 34 — UML-диаграмма понятия онтологии повышенного уровня

Примечание — Понятие представления свойства определено в разделе 11.15.1 ИСО 13584-24:2003.

Определение внешнего типа:

Тип **PROPERTY_Type:** См. 6.7.4.

6.7.6 Тип идентифицированных данных

В определенном контексте оказывается полезным определять область значений, которая связана с глобальным идентификатором и которая может повторно использоваться для нескольких свойств (возможно, даже в нескольких онтологиях).

Пример — Онтология, определяющая специфические для области значений единицы, может основываться на использовании онтологических типов данных.

Для этой цели OntoML-язык предлагает комплексный XML-тип данных **DATATYPE_Type** (см. рисунок 35).

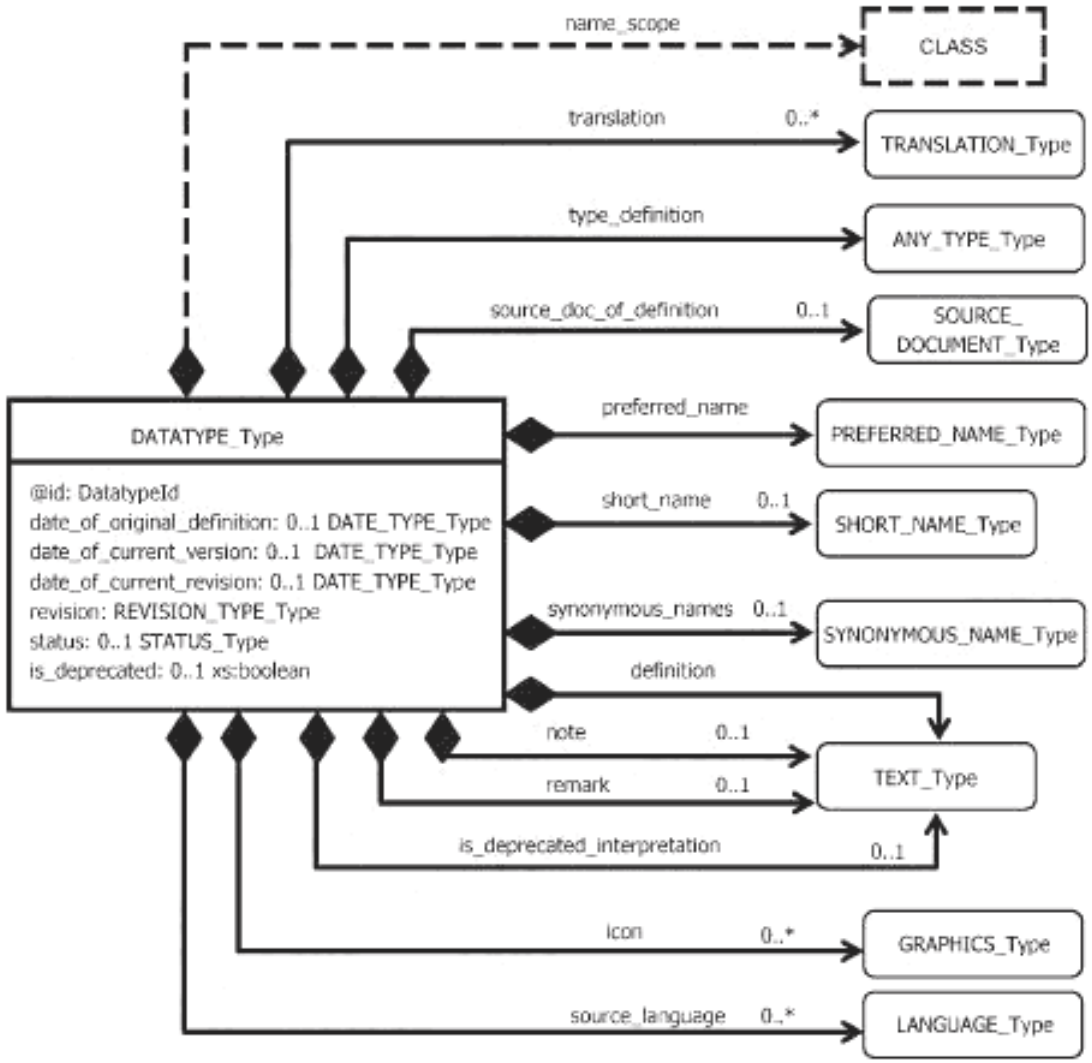


Рисунок 35 — UML-диаграмма типов данных

Определения внутренних элементов:

Элемент **@id:** Определяет идентификатор типа данных.

Элемент **date_of_current_revision:** Определяет дату, связанную с существующей редакцией определения типа данных.

Элемент **date_of_current_version:** Определяет дату, связанную с существующей версией определения типа данных.

Элемент **date_of_original_definition:** Определяет дату, связанную с первой неизменной версией определения типа данных.

Элемент **definition:** Определяет текст, описывающий данный тип данных (возможно, переведенный).

Элемент **icon:** Определяет пиктограммы, иллюстрирующие описание и связанные с именами.

Элемент **is_deprecated:** Определяет булев элемент, который указывает (если он истинен), что данное определение типа данных не должно больше использоваться.

Элемент **is_deprecated_interpretation:** Определяет способ интерпретации причины исключения, примерные значения от поставщика и соответствующий идентификатор.

Элемент **name_scope:** Определяет область класса данного свойства.

Элемент **note:** Определяет дополнительную информацию относительно любой части свойства, которая существенна для его понимания (возможно, переведенную).

Элемент **preferred_name:** Определяет имя свойства, которое является предпочтительным для его использования (возможно, переведенное).

Элемент **remark:** Определяет пояснительный текст, дополнительно разъясняющий содержание данного свойства (возможно, переведенный).

Элемент **revision:** Определяет номер версии существующего определения свойства.

Элемент **short_name:** Определяет сокращение предпочтительного имени свойства (возможно, переведенное).

Элемент **source_doc_of_definition:** Определяет возможный документ-источник, из которого данное определение было заимствовано.

Элемент **source_language:** Определяет язык, на котором первоначально было дано описание класса и на котором сохраняется исходное содержание в случае несоответствия перевода.

Элемент **status:** Определяет состояние в жизненном цикле определения класса.

Примечание 1 — Допускаемые коды состояния определяются путем заключения частного соглашения между поставщиком словаря и его пользователями.

Примечание 2 — Если XML-элемент состояния не предоставляется и если данное определение свойства не исключается как обозначенное с помощью возможного XML-элемента **is_deprecated**, то это определение свойства будет иметь то же состояние стандартизации, что и в целом у используемой онтологии. В частности, если онтология стандартизирована, то это свойство будет являться частью текущей редакции стандарта.

Элемент **synonymous_names:** Определяет набор синонимических имен предпочтительных символов свойства.

Элемент **translation:** Определяет возможный массив переведенной информации, предоставляемый для переводимых элементов.

Элемент **type_definition:** Определяет описание типа, предназначенного для передачи данных.

Определения внутренних типов:

Тип **DATE_TYPE_Type:** Является идентификатором значений, допустимых для обозначения даты (**xs:date** XML-схемы типов данных).

Тип **REVISION_TYPE_Type:** Является строкой (**xs:string** XML-схемы типов данных), представляющей значения, допускаемые для обозначения редакции. Эта строка должна содержать не более трех символов.

Тип **STATUS_Type:** Является строкой (**xs:string** XML-схемы типов данных), представляющей значения, допускаемые для обозначения состояния.

Определения внешних типов:

Элемент **DatatypeId:** См. 9.1.

Тип **ANY_TYPE_Type:** См. 8.3.

Тип **GRAPHICS_Type:** См. 8.2.2.2.

Тип **LANGUAGE_Type:** См. 8.1.1.

Тип **PREFERRED_NAME_Type:** См. 8.1.2.

Тип **SHORT_NAME_Type:** См. 8.1.2.

Тип **SOURCE_DOCUMENT_Type:** См. 8.2.2.1.

Тип **SYNONYMOUS_NAME_Type:** См. 8.1.2.
Тип **TEXT_Type:** См. 8.1.2.
Тип **TRANSLATION_Type:** См.8.1.3.
Перечень ограничительных условий:
Если элемент **is_deprecated** существует, то должен существовать и элемент **is_deprecated_interpretation**.
Примерные значения элемента **is_deprecated_interpretation** должны определяться в тот момент, когда принималось решение о возражении.

6.7.7 Документ

В OntoML-языке документ может связываться с глобальным идентификатором и рассматривается как понятие CIIM-онтологии. Для этой цели OntoML-язык предлагает комплексный XML-тип данных **DOCUMENT_Type** (см. рисунок 36).

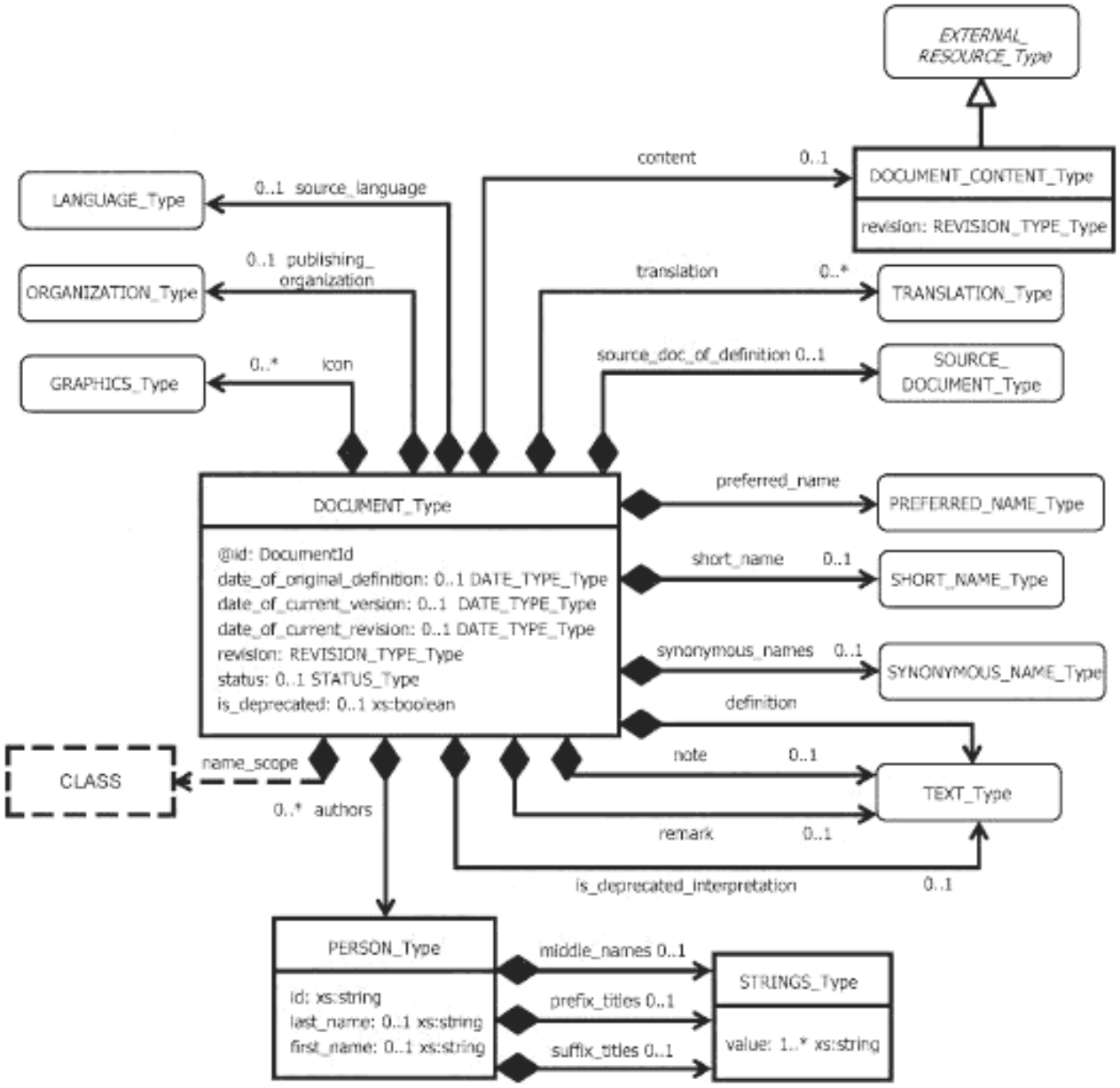


Рисунок 36 — UML-диаграмма документа простого уровня

Определения внутренних элементов:

Элемент **@id:** Определяет идентификатор документа.

Элемент **authors:** Определяет автора (авторов) документа.

Элемент **authors/first_name:** Определяет первый элемент списка личных имен сотрудников.

Элемент **authors/id:** Определяет способ идентификации сотрудника.

Элемент **authors/last_name:** Определяет фамилию сотрудника.

Элемент **authors/middle_names:** Определяет другие фамилии сотрудников (если их несколько).

Элемент **authors/prefix_titles:** Определяет слово (или группу слов), которое определяет социальное и/или профессиональное положение сотрудника и указывается перед его/ее фамилией.

Элемент **authors/suffix_titles:** Определяет слово (или группу слов), которое определяет социальное и/или профессиональное положение сотрудника и указывается после его/ее фамилии.

Элемент **authors/middle_names/value:** Определяет строку со вторым именем в строковой совокупности.

Элемент **authors/prefix_titles/value:** Определяет строку префиксного названия в строковой совокупности.

Элемент **authors/suffix_titles/value:** Определяет строку суффиксного названия в строковой совокупности.

Элемент **content:** Определяет физический документ, для которого комплексный XML-тип данных **DOCUMENT_Type** обеспечивает описание.

Примечание 1 — Содержание документа представляется комплексным XML-типом данных **DOCUMENT_CONTENT_Type** и определяется как подтип комплексного XML-типа данных **EXTERNAL_RESOURCE_Type** согласно 8.2, поэтому наследованный XML-элемент файла позволяет давать ссылку на намеченный документ с помощью унифицированного идентификатора ресурса (URI).

Примечание 2 — Физический документ является вспомогательным и может (или не может) предоставляться в виде того же экземпляра OntoML-документа.

Элемент **content/revision:** Определяет характеристики обновления физического документа.

Элемент **date_of_current_revision:** Определяет дату, связанную с существующим вариантом определения документа.

Элемент **date_of_current_version:** Определяет дату, связанную с существующей версией определения документа.

Элемент **date_of_original_definition:** Определяет дату, связанную с первой неизменной версией определения документа.

Элемент **definition:** Определяет текст, описывающий данный документ (возможно, переведенный).

Элемент **icon:** Определяет графические материалы, представляющие описания, которые связаны с именами.

Элемент **is_deprecated:** Определяет булев элемент (если он истинен), который устанавливает, что определение документа не должно больше использоваться.

Элемент **is_deprecated_interpretation:** Определяет способ интерпретации причины возражения, примерные значения от возражающего поставщика и соответствующий идентификатор.

Элемент **name_scope:** Определяет область определения класса документа.

Элемент **note:** Определяет дополнительную информацию к любой части документа, которая существенна для понимания этой информации (возможно, переведенной).

Элемент **preferred_name:** Определяет имя документа, которое является предпочтительным для его использования (возможно, переведенное).

Элемент **publishing_organization:** Определяет организацию, которая публикует документ.

Элемент **remark:** Определяет пояснительный текст, дополнительно проясняющий содержание данного документа (возможно, переведенного).

Элемент **revision:** Определяет номер редакции определения документа.

Элемент **short_name:** Определяет сокращенное наименование документа (возможно, переведенное).

Элемент **source_doc_of_definition:** Определяет предполагаемый документ-источник, из которого было заимствовано определение.

Элемент **source_language:** Определяет язык, на котором первоначально было дано описание класса и на котором сохраняется исходное содержание в случае несоответствия перевода.

Элемент **status:** Определяет состояние определения документа в его жизненном цикле.

Примечание 3 — Допустимые значения для состояния определяются путем частного соглашения между поставщиком словаря и пользователями этого словаря.

Примечание 4 — Если XML-элемент состояния не предоставляется и если данное определение свойства не исключается как обозначенное с помощью возможного XML-элемента **is_deprecated**, то это определение свойства будет иметь то же состояние стандартизации, что и в целом у используемой онтологии. В частности, если онтология стандартизирована, то это свойство будет являться частью текущей редакции стандарта.

Элемент **synonymous_names:** Определяет набор синонимических имен предпочтительных символов свойства.

Элемент **translation:** Определяет возможный массив переведенной информации, предоставляемый для переводимых элементов.

Определения внутренних типов:

Тип **DATE_TYPE_Type:** Является идентификатором значений, допустимых для обозначения даты (**xs:date** XML-схемы типов данных).

Тип **DOCUMENT_CONTENT_Type:** Является физическим ресурсом, с которым связано определение документа.

Тип **REVISION_TYPE_Type:** Является строкой (**xs:string** XML-схемы типов данных), которая представляет значения, допускаемые для обозначения редакции. Эта строка должна содержать не более трех символов.

Тип **STATUS_Type:** Является строкой (**xs:string** XML-схемы типов данных), представляющей значения, допускаемые для обозначения состояния.

Тип **STRINGS_Type:** Является строкой (**xs:string** XML-схемы типов данных), представляющей ее хранилище.

Определения внешних типов:

Элемент **DatatypeId:** См. 9.1.

Тип **EXTERNAL_RESOURCE_Type:** См. 8.2.1.

Тип **GRAPHICS_Type:** См. 8.2.2.2.

Тип **LANGUAGE_Type:** См. 8.1.1.

Тип **ORGANIZATION_Type:** См. 8.8.1.

Тип **PREFERRED_NAME_Type:** См. 8.1.2.

Тип **SHORT_NAME_Type:** См. 8.1.2.

Тип **SOURCE_DOCUMENT_Type:** См. 8.2.2.1.

Тип **SYNONYMOUS_NAME_Type:** См. 8.1.2.

Тип **TEXT_Type:** См. 8.1.2.

Тип **TRANSLATION_Type:** См. 8.1.3.

Перечень ограничительных условий:

Если элемент **is_deprecated** существует, то должен существовать и элемент **is_deprecated_interpretation**.

Примерные значения элемента **is_deprecated_interpretation** должны определяться в тот момент, когда принималось решение о возражении.

7 Общие сведения о представлении OntoML-библиотек

Содержащая библиотеку часть OntoML-языка предоставляет ресурсы представления экземпляров, принадлежащих классам, которые определены в заданной области онтологии. Подобные экземпляры могут быть характеристиками продукции, если они принадлежат к классу характеристик, или представлениями продукции, если они принадлежат к классу функциональных моделей.

Примечание 1 —Продукция понимается в очень широком, обобщенном смысле, распространяющимся на элементы, которые могут характеризоваться с помощью OntoML-классов элементов.

Примечание 2 — Содержание библиотеки может связываться или не связываться с онтологией, описанной в экземпляре OntoML-документа.

Примечание 3 — Содержание библиотеки предоставляется, в частности, для обмена электронными каталогами.

7.1 Корневой элемент библиотеки

В OntoML-языке каждая часть библиотечной информации соединяется в общую структуру, которая является комплексным XML-типом данных **LIBRARY_TYPE** (см. рисунок 37).

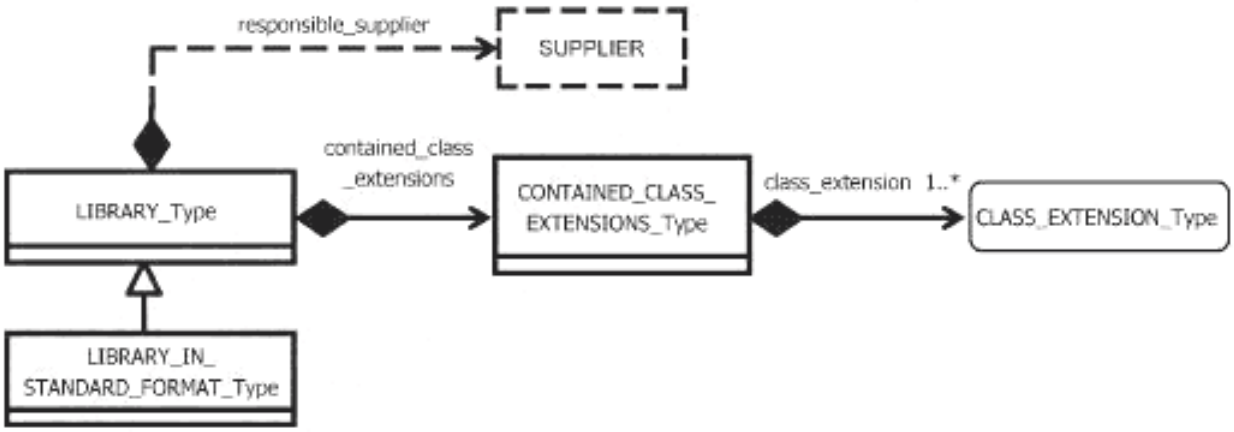


Рисунок 37 — Корневой элемент библиотеки

Определения внутренних элементов:

Элемент **contained_class_extensions:** Определяет множество расширений класса для множества онтологических классов.

Элемент **contained_class_extensions/class_extension:** Определяет расширение класса, содержащееся в словаре.

Элемент **responsible_supplier:** Определяет поставщика информации, ответственного за содержание библиотеки.

Определения внутренних типов:

Тип **CONTAINED_CLASS_EXTENSIONS_Type:** Является последовательностью описаний расширений класса.

Тип **LIBRARY_IN_STANDARD_FORMAT_Type:** Является библиотекой, которая использует только протоколы внешних файлов и допускается либо встроенной в библиотеку информационной моделью (индицируемой с помощью XML-элемента **library_structure**), либо протоколами обмена представлениями (на которые дана ссылка в XML-элементе **supported_vep**), причем в обоих случаях определенных комплексным XML-типом данных **HEADER_Type**.

Тип **LIBRARY_Type:** Является хранилищем для представления различных частей библиотечной информации.

Определение внешнего типа:

Тип **CLASS_EXTENSION_Type:** См. раздел 7.2.

7.2 Общая структура расширений класса

Расширение класса представляется с помощью абстрактного комплексного XML-типа данных **CLASS_EXTENSION_Type** (см. рисунок 38), который позволяет определять, в частности, для любых видов расширений класса, описывается ли каждый экземпляр одними и теми же свойствами в одном и том же порядке, имея вид строки таблицы (XML-элемент **table_like**), а также позволяет ли определять множество применимых к классу свойств, которые необходимы и достаточны для идентификации каждого экземпляра, принадлежащего к расширению класса. Таким образом, в случае таблицы подобной структуры содержания он будет соответствовать ключу к этой таблице и выполняться с помощью XML-элемента **instance_identification** (см. рисунок 38).

Примечание 1 — Свойства, которые соответствуют табличному ключу, должны связываться со значениями для всех экземпляров класса. Это определяется в спецификации на ограничительные условия для подкласса.

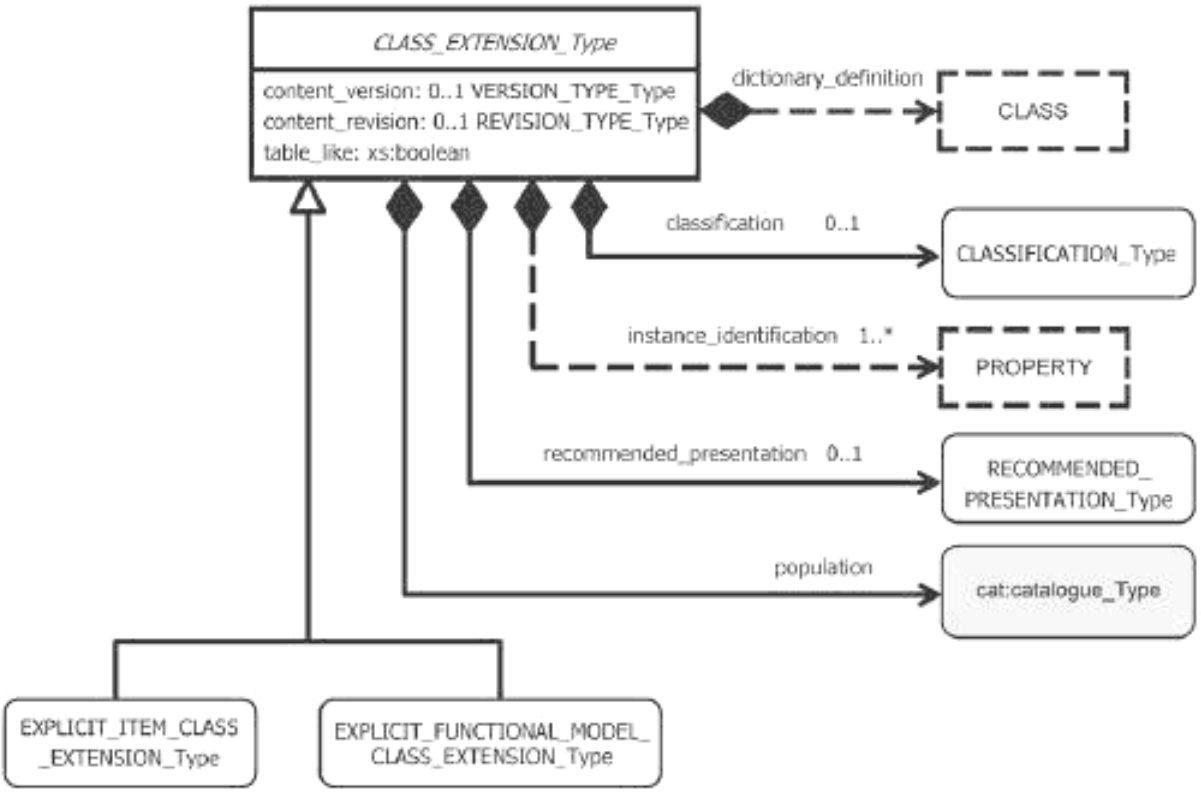


Рисунок 38 — Обобщенная структура класса расширения

Определения внутренних элементов:

Элемент **classification:** Определяет возможную ссылку на классификацию свойств, используемых для описания экземпляров класса.

Элемент **content_revision:** Определяет номер редакции, который соответствует текущему описанию варианта элемента **content_version** расширения класса.

Элемент **content_version:** Определяет номер версии, который характеризует расширение класса, т. е. множество допускаемых экземпляров класса.

Элемент **dictionary_definition:** Определяет ссылку на определение расширения класса в словаре.

Элемент **instance_identification:** Определяет ссылки на свойства, которые позволяют однозначно идентифицировать каждый экземпляр, принадлежащий какому-либо классу.

Элемент **population:** Определяет список экземпляров класса, которые описывают семейство классов.

Элемент **recommended_presentation:** Определяет рекомендуемый коэффициент масштабирования, единицы представления информации и форматы представления значений, которые должны использоваться для индикации значений ссылочных свойств в контексте ссылочного класса.

Элемент **table_like:** Определяет булево значение, которое определяет, характеризуются ли все экземпляры класса одними и теми же свойствами и в одно и том же порядке, или нет.

Определения внутренних типов:

Тип **CLASS_EXTENSION_Type:** Является абстрактным комплексным XML-типом данных, супертипом различных расширений класса.

Тип **REVISION_TYPE_Type:** Является строкой (**xs:string** XML-диаграммы), представляющей значения, которые допускаются для данной редакции. Эта строка не должна содержать более трех символов.

Тип **VERSION_TYPE_Type:** Является строкой (**xs:string** XML-диаграммы), представляющей значения, которые допускаются для данного варианта. Эта строка должна содержать только цифры, число которых не должно превышать 9.

Определения внешних типов:

Тип **cat:catalogue_Type:** Является перечнем экземпляров класса как множества ссылок на свойства и пары значений.

Примечание 2 — Тип **cat:catalogue_Type** определен в ИСО/ТС 29002-10 на коммуникативный формат обмена данными.

Тип **CLASSIFICATION_Type:** См. 7.2.1.
Тип **EXPLICIT_FUNCTIONAL_MODEL_CLASS_EXTENSION_Type:** См. 7.4.
Тип **EXPLICIT_ITEM_CLASS_EXTENSION_Type:** См. 7.3.
Тип **RECOMMENDED_PRESENTATION_Type:** См. 7.2.2.
Перечень ограничительных условий:

Каждый экземпляр класса, определяющий семейство класса, на которое даны ссылки в совокупности элементов **instance_identification,** никогда не должен быть связан с нулевым значением.

Свойства в версиях одного и того же класса, на которые даются ссылки в семействе элементов **instance_identification,** не должны изменяться.

В одном и том же классе любой его версии одни и те же значения элемента **instance_identification** свойства должны соответствовать той же части.

Элементы **content_version** и **content_revision** должны существовать совместно.

Элемент **content_version** должен увеличиваться тогда и только тогда, когда расширение класса изменяется (становятся допустимыми новые экземпляры класса), либо предыдущие экземпляры класса в дальнейшем становятся недопустимыми.

При возрастании элемента **content_version** элемент **version** связанного с ним определения класса (на который дается ссылка с помощью XML-элемента **dictionary_definition**) не обязательно будет возрастать.

Элемент **content_revision** должен увеличиваться при любом изменении описания расширения класса, однако изменения будут модифицировать допустимые экземпляры в данном классе.

При возрастании элемента **content_version** элемент **content_revision** должен сбрасываться на нуль.

Если XML-элемент **table_like** является истинным, то свойства, которые описывают каждый экземпляр класса, должны иметь те же свойства в том же порядке.

Все свойства, на которые даны ссылки в совокупности элементов **instance_identification,** должны быть применимыми к данному классу. Элемент **instance_identification** не должен содержать дублированные ссылки на свойство.

Все свойства, используемые для определения любого экземпляра качества (посредством XML-элемента **population**), должны быть применимы к данному классу.

Все экземпляры класса, которые определяют класс элементов **population,** должны быть такими же, как все ссылочные элементы **property_values,** чьими значениями являются элементы **translated_string_values,** и переведены на один и тот же язык (языки).

Все экземпляры класса, которые определяют элементы класса **population,** должны ссылаться на один и тот же класс, а не на класс, на который дается ссылка с помощью элемента **explicit_model_class_extension** (посредством его унаследованного XML-элемента **dictionary_definition**).

7.2.1 Классификация свойств

Некоторые свойства, входящие в описания экземпляров класса, могут быть сгруппированы с помощью некоторых классификационных ресурсов. Каждая группа идентифицируется с помощью целого числа и предназначена для различной обработки на приемной системе информации (свойств), принадлежащей данной группе.

Примечание 1 — Свойство может принадлежать различным группам.

Примечание 2 — Свойство, которое не связано с классификационным значением, не связано с какой-либо конкретной обработкой.

Примечание 3 — В настоящем стандарте не сделано никакого предположения относительно интерпретации каждого классификационного значения на приемной системе, в результате чего может заключаться частное соглашение между отправителем и получателем.

Классификация свойств представляется с использованием комплексного XML-типа данных **CLASSIFICATION_Type** (см. рисунок 39).

Рисунок 39 — Классификация свойств

Определения внутренних элементов:

Элемент **property_classification:** Определяет классификацию свойств.

Элемент **property_classification/its_value:** Определяет классификационное значение, связанное с ссылочным элементом **prop_def**.

Элемент **property_classification/prop_def**: Определяет ссылку на свойство, которое классифицируется с помощью значения элемента **its_value**.

Определения внутренних типов:

Тип **CLASSIFICATION_Type:** Является хранилищем классификации свойств.

Тип **PROPERTY_CLASSIFICATION_Type:** Является связью, которая закрепляет классификационную группу за конкретным свойством.

7.2.2 Представление свойств

Когда для описания экземпляров в конкретном классе используется конкретное свойство, может оказаться полезным использовать особый масштабный коэффициент, формат отображения единицы и/или их значений. Подобный вид рекомендации представляется с использованием комплексного XML-типа данных **RECOMMENDED_PRESENTATION_Type** (см. рисунок 40).

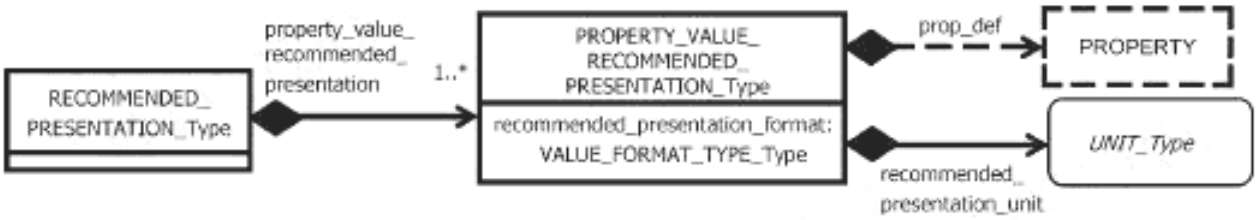


Рисунок 40 — Представление свойств

Определения внутренних элементов:

Элемент **property_value_recommended_representation:** Определяет свойства рекомендованных представлений.

Элемент **property_value_recommended_representation/prop_def:** Определяет ссылку на свойство, которое поставщик библиотеки данных рекомендует для преобразования значений с целью представления.

Элемент **property_value_recommended_representation/recommended_presentation_format:** Определяет формат представления информации, рекомендуемый поставщиком библиотеки данных для представления значений ссылочного элемента **prop_def,** при условии, что эти значения преобразуются в единицы элемента **recommended_presentation_unit**.

Элемент **property_value_recommended_representation/recommended_presentation_unit:** Определяет конкретную единицу, с помощью которой поставщик библиотечных данных рекомендует преобразовывать данные для целей представления.

Определения внутренних типов:

Тип **RECOMMENDED_PRESENTATION_Type:** Является хранилищем рекомендуемого представления свойства.

Тип **PROPERTY_VALUE_RECOMMENDED_PRESENTATION_Type:** Является форматом рекомендуемого представления свойства вместе с соответствующей единицей представления.

Тип **VALUE_FORMAT_TYPE_Type:** Является идентификатором значений, допускаемым для формата этих значений.

Примечание — Значения типа **VALUE_FORMAT_TYPE_Type** определены согласно приложению Н.

Определение внешнего типа:

Тип **UNIT_Type:** Является спецификацией единицы, см. 8.4.

Перечень ограничительных условий:

Длина значения типа **VALUE_FORMAT_TYPE_Type** не должна превышать 80 символов.

Единица, определенная с помощью XML-элемента **recommended_presentation_unit,** должна быть совместима с основным типом данных, связанным со свойством, на который дается ссылка с помощью XML-элемента **prop_def**.

7.3 Библиотека простого уровня: содержание классов продукции

Описание продукции, принадлежащей к заданному классу характеристик продукции, производится с использованием комплексного XML-типа **EXPLICIT_ITEM_CLASS_EXTENSION_Type** (см. рисунок 41). Он

также предоставляет дополнительную информацию, которую можно использовать на приемной системе для отображения содержания класса характеристик продукции.

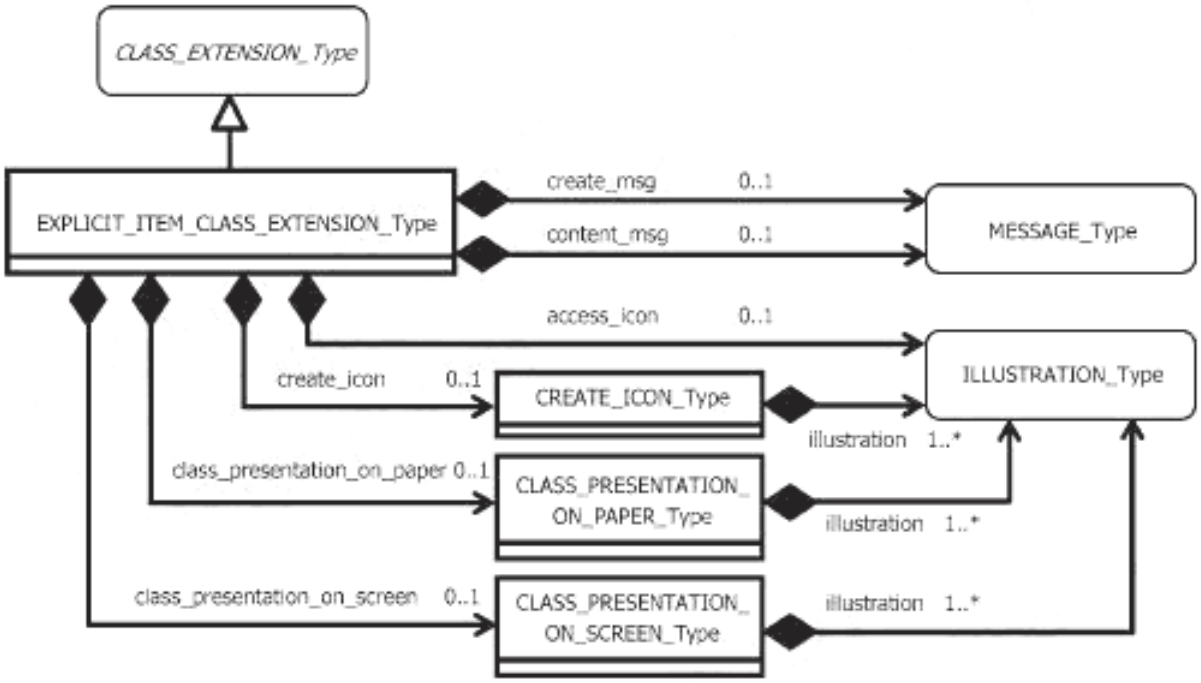


Рисунок 41 — Структура представления характеристик продукции

Определения внутренних элементов:

Элемент **access_icon:** Определяет дополнительную пиктограмму, которая позволяет представлять класс в меню.

Примечание 1 — Пиктограммы определяются как изображение формата А9 стандартизированного размера.

Элемент **class_presentation_on_paper:** Определяет упорядоченный набор иллюстраций, рекомендованный поставщиком библиотеки данных, который должен представляться пользователю, когда содержание класса представляется на бумаге, например, для печати этого содержания в виде буклета.

Элемент **class_presentation_on_paper/illustration:** Определяет класс иллюстраций, который должен представляться на бумаге, например, для печати этого содержания в виде буклета.

Элемент **class_presentation_on_screen:** Определяет упорядоченный набор иллюстраций, рекомендованный поставщиком библиотеки данных, который должен представляться пользователю, когда содержание класса представляется на экране.

Элемент **class_presentation_on_screen/illustration:** Определяет класс иллюстраций, которые должны отображаться на экране.

Элемент **content_msg:** Определяет сообщение, включающее в себя содержание класса.

Элемент **create_icon:** Определяет дополнительную пиктограмму (пиктограммы), которая обеспечивает визуальное представление свойств элементов класса и эталонной системы координат.

Примечание 2 — Пиктограммы определяются как изображение формата А9 стандартизированного размера.

Элемент **create_icon/illustration:** Определяет пиктограмму-иллюстрацию, которая должна отображаться на экране.

Элемент **create_msg:** Определяет дополнительное сообщение, описывающее свойства элементов класса и эталонную систему координат.

Определения внутренних типов:

Тип **CLASS_PRESENTATION_ON_PAPER_Type:** Является структурой бумажной иллюстрации.

Тип **CLASS_PRESENTATION_ON_SCREEN_Type:** Является структурой экранной иллюстрации.

Тип **CREATE_ICON_Type:** Является визуальным представлением свойств.

Определения внешних типов:

Тип **CLASS_EXTENSION_Type:** См. 7.2.

Тип **ILLUSTRATION_Type:** См. 8.2.1.2.

Тип **MESSAGE_Type:** См. 8.2.1.3.

Перечень ограничительных условий:

Класс, на который дается ссылка с помощью наследуемого XML-элемента **dictionary_definition**, должен иметь в качестве основного типа тип **ITEM_CLASS_Type** или **ITEM_CLASS_CASE_OF_Type**.

Каждый элемент **illustration**, определенный в совокупности элементов **class_presentation_on_paper illustration**, должен устанавливать значения для XML-элементов **width** и **height**, но не значение элемента **not_static_picture** для XML-элемента **kind_of_content**.

Каждый элемент **illustration**, определенный в элементе **class_presentation_on_screen illustrations**, должен устанавливать значения для XML-элементов **width** и **height**.

7.4 Библиотека повышенного уровня: содержание классов представления продукции

Каждый экземпляр класса функциональных моделей, называемый «функциональной моделью», является представлением продукции, которое содержит список пар «свойство/значение». Подкласс этих свойств, которые входят в наследованный XML-элемент **instance_identification**, составляет ядро этих экземпляров. Элемент **instance_identification** содержит всю необходимую информацию для идентификации одного экземпляра класса и ограничения его от других экземпляров того же класса. Это множество экземпляров класса регистрируется в наследованном XML-элементе **population**.

Примечание 1 — Каждая функциональная модель является моделью одного (или нескольких) вида продукции производного (view of) класса. Соединение выполняется с помощью свойства (свойств) продукции, ссылка на которое дается в XML-элементе **required_item_values**. Эти свойства должны быть свойствами, импортируемыми из элемента, и они должны дублироваться в классе функциональных моделей и производном классе элементов. Кроме того, эти свойства позволяют устанавливать связь между классом элементов и классом функциональных моделей.

При отсутствии контрольных переменных представления, определенных в классе функциональных представлений, ссылка на который дается с помощью класса функциональных моделей, код класса функциональных моделей, определенный с помощью наследуемого атрибута **instance_identification**, идентичен группе свойств, ссылка на которую дается в XML-элементе **required_item_values**, поэтому оператор связи связывается с каждым экземпляром элемента как минимум одной функциональной модели.

Если некоторые контрольные переменные представления определены в классе функциональных представлений, ссылка на который дается с помощью класса функциональных моделей, код класса функциональных моделей, определенный с помощью наследуемого атрибута **instance_identification**, идентичен объединенной группе свойств, ссылка на которую дается в XML-элементе **required_item_values**, или группе свойств, ссылка на которую дается в XML-элементе **view_control_variables** класса функциональных представлений, поэтому оператор связи связывается с каждым экземпляром элемента как минимум одной функциональной модели для каждой записи значений множества контрольных переменных представления.

Пример 1 — Предположим, что уникальный производный (view-of) класс функциональных моделей определен для корневого узла класса элементов библиотеки. Предположим также, что в этом классе функциональных моделей определено свойство inventory status каждой продукции в каждом классе элементов. Класс функциональных моделей может быть описан с помощью трех свойств: the part number (импортированного из класса корневых элементов), the stock availability и quantity of order. Если XML-элемент required_item_values производного (view-of) класса функциональных моделей приводит ссылку только на свойство part_number, то при выборе продукции состояние ее запасов может рассчитываться автоматически. Кроме того, множество данных о состоянии запасов всей продукции может также рассчитываться с помощью системы посредством связи между расширением этого класса элементов и расширением класса функциональных моделей.

Пример 2 — Если класс функциональных моделей дает представления, соответствующие значениям нескольких контрольных переменных представления для каждой продукции, то пользователю необходимо выбирать конкретное значение для каждой контрольной переменной представления с целью выбора требуемого представления этой продукции.

Расширения класса функциональных моделей представляются с помощью комплексного XML-типа данных **FUNCTIONAL_MODEL_CLASS_EXTENSION_Type** (см. рисунок 42).

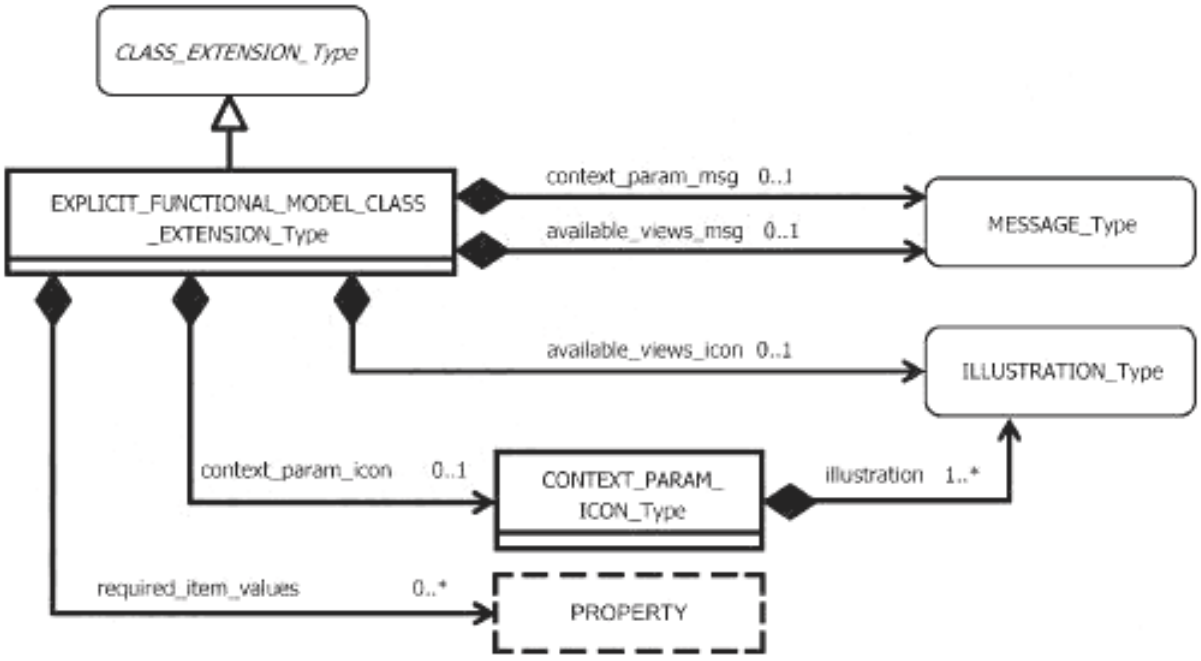


Рисунок 42 — UML-диаграмма структуры функциональных моделей

Определения внутренних элементов:

Элемент **available_views_icon:** Определяет предполагаемые иллюстрации, которые способны давать визуальное представление в различных видах и которые могут создаваться с помощью класса функциональных моделей.

Примечание 2 — Пиктограммы определены в стандартизированном размере формата А6.

Элемент **available_views_msg:** Определяет возможные сообщения, которые описывают различные виды и могут создаваться с помощью класса функциональных моделей.

Элемент **context_param_icon:** Определяет возможные пиктограммы, которые могут давать визуальное представление свойств и должны присваивать значения конкретному выбранному виду.

Элемент **context_param_icon/illustration:** Определяет иллюстрации, которые представляют элемент **context_param_icon**.

Примечание 3 — Иллюстрации определены в стандартизированном размере формата А6.

Элемент **context_param_msg:** Определяет возможные сообщения, поясняющие свойства модели, которые должны присваивать значения конкретному выбранному виду.

Элемент **required_item_values:** Определяет характеристики элемента, используемые для согласования экземпляров класса функциональных моделей с экземплярами класса элементов.

Примечание 4 — Свойства, на которые даны ссылки в группе элементов **required_item_values,** должны представляться и в классе элементов, и в классе функциональных моделей. После этого согласование осуществляется с помощью внешней связи этих классов.

Примечание 5 — Должны появляться только те характеристики элементов, которые необходимы для создания экземпляров класса функциональных моделей.

Примечание 6 — Если класс функциональных моделей не является частным производным (view-of) классом элементов, то группа элементов **required_item_values** пуста, а если этот класс соединяется с классом элементов в системе пользователя, то для связи класса функциональных моделей с этим классом элементов должен использоваться комплексный XML-тип данных **A_POSTERIORI_VIEW_OF_Type** (см. 8.6.2), также как и для определения того, какие свойства класса функциональных моделей должны связываться со свойствами класса элементов при установлении связи между этими классами.

Определение внутреннего типа:

Тип **CONTEXT_PARAM_ICON_Type:** Является структурой пиктограммы контекстного параметра.

Определение внешнего типа:

Тип **CLASS_EXTENSION_Type:** См. 7.2.

Перечень ограничительных условий:

Класс, ссылка на который дается с помощью наследуемого XML-элемента **dictionary_definition,** должен быть либо классом с типом данных **FUNCTIONAL_MODEL_CLASS_Type,** либо классом с типом данных **FM_CLASS_VIEW_OF_Type**.

Если группа элементов **required_item_values** не является пустой, то наследуемый XML-элемент **dictionary_definition** должен иметь как основной тип данных **FM_CLASS_VIEW_OF_Type,** и каждое свойство, на которое дается ссылка в группе элементов **required_item_values,** должно быть свойством с типом данных **NON_DEPENDENT_P_DET_Type**.

Если группа элементов **required_item_values** не является пустой, то наследуемый XML-элемент **dictionary_definition** должен иметь основной тип данных **FM_CLASS_VIEW_OF_Type,** и каждое свойство, на которое дается ссылка в группе элементов **required_item_values,** должно принадлежать группе элементов **imported_properties_from_item** данного ссылочного класса с типом данных **FM_CLASS_VIEW_OF_Type**.

Каждое свойство, на которое дается ссылка в группе элементов **required_item_values,** также должно принадлежать множеству свойств, на которые дается ссылка в группе элементов **instance_identification**.

Каждый экземпляр класса, принадлежащий группе элементов **population,** должен описываться с помощью всех контрольных переменных представления, которые определяются в классе функциональных представлений (XML-элемент **created_view**), ссылка на который дается с помощью связанного класса функциональных моделей (наследуемый XML-элемент **dictionary_definition**).

Группа наследуемых элементов **instance_identification** должна относиться как ко всем свойствам, используемым для представления контрольных переменных, которые определены в ссылочном классе функциональных представлений (XML-элемент **created_view**) с помощью связанного с ним класса функциональных моделей (наследуемый XML-элемент **dictionary_definition**), так и к свойствам, на которые дается ссылка в группе элементов **required_item_values**.

8 Прочие структурированные информационные элементы

В данном разделе определены конструкции прочих структурированных информационных элементов, на которые были даны ссылки в разделе 7.

8.1 Переводы

OntoML-язык обеспечивает ресурсы для перевода нешифрованной текстовой информации и управления переводом.

8.1.1 Спецификация языка

Спецификация языка двойственна: собственно спецификация и, возможно, связанная с ней страна, которая в дальнейшем будет определять этот язык (см. рисунок 43):

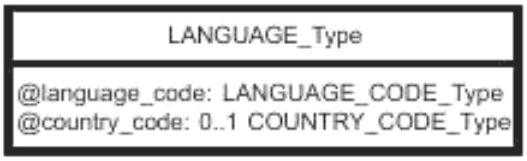


Рисунок 43 — Спецификация языка

Определения внутренних элементов:

Элемент **@country_code:** Определяет возможный код страны, которая в дальнейшем будет определять лингвистический код.

Элемент **@language_code:** Определяет код языка.

Определения внутренних типов:

Тип **COUNTRY_CODE_Type:** Является типом лингвистического кода и является строкой, содержащей два символа и определяющей страну согласно ИСО 3166-1.

Тип **LANGUAGE_CODE_Type:** Является типом лингвистического кода и является строкой, содержащей два-три символа и определяющей язык согласно ИСО 639-1 и ИСО 639-2, соответственно.

8.1.2 Перевод выражаемых строкой элементов

Каждое понятие CIIM-онтологии описывается с использованием нешифрованной текстовой информации, которая может или не может переводиться. Соответствующие информационные элементы таковы:

- Элемент **preferred_names;**

- Элемент **short_names**;
- Элемент **synonymous_names**;
- Элемент **keywords**;
- Элементы **definitions, notes and remarks**;
- Элемент **source_doc_of_definition** (если он определен как идентификатор документа, см. 8.2.2.1.1).

OntoML-язык дает конструкции для представления этих информационных элементов (см. рисунок 44).

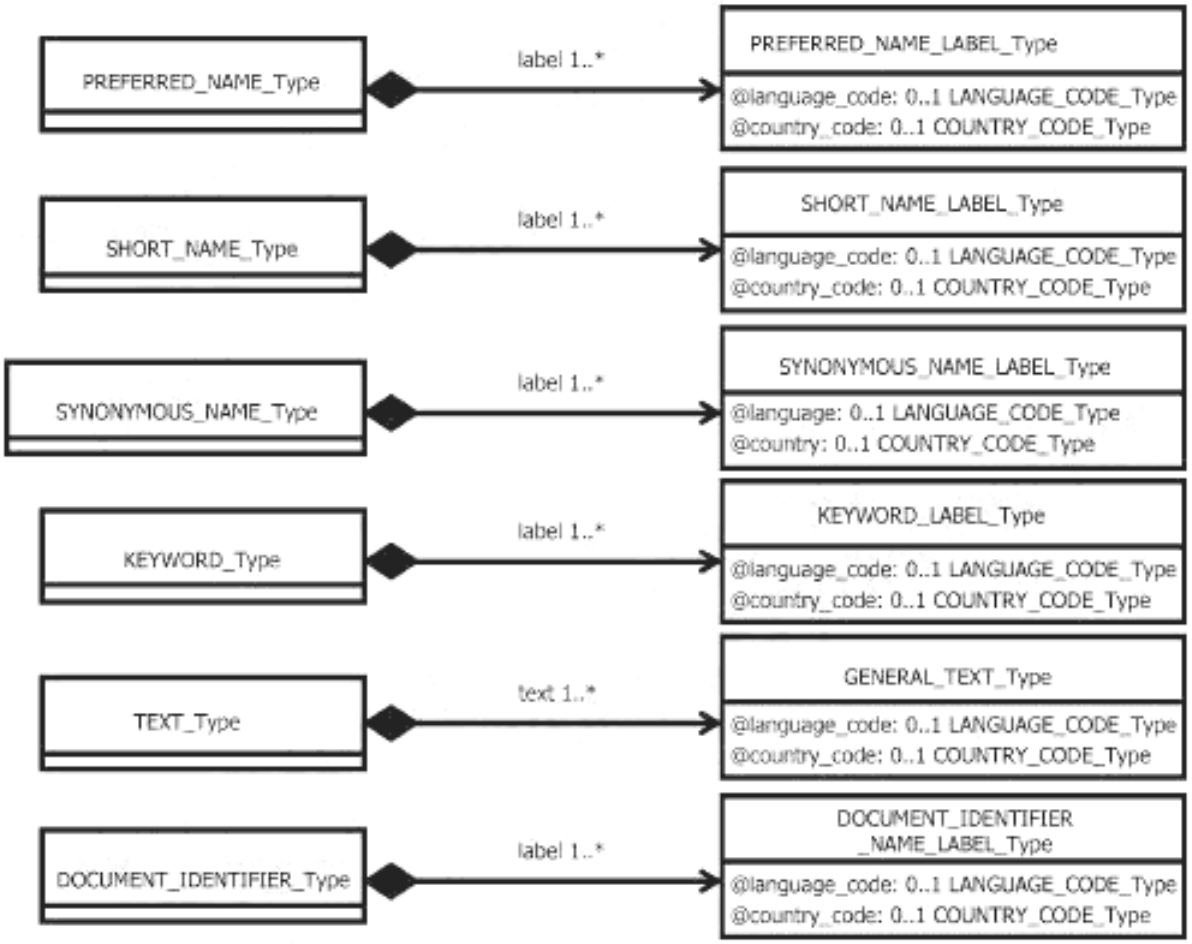


Рисунок 44 — Трансляционные ресурсы

Примечание 1 — Минимальное требование к лингвистической информации состоит в предоставлении значения для XML-атрибута **language_code**.

Примечание 2 — Если язык онтологии определен с использованием XML-элемента **global_language**, определенного в комплексном типе данных **HEADER_Type**, то ни XML-атрибут **language_code**, ни атрибут **country_code** не должны предоставляться.

Примечание 3 — Тип **HEADER_Type** определен в 6.5.

Определения внутренних элементов:

Элемент **label (DOCUMENT_IDENTIFIER_Type):** Определяет метки документов, которые, возможно, были переведены вместе с их соответствующими переводами.

Элемент **label (KEYWORD_Type):** Определяет метки ключевых слов, которые, возможно, были переведены вместе с их соответствующими переводами.

Элемент **label (PREFERRED_NAME_Type):** Определяет метки предпочитаемых имен, которые, возможно, были переведены вместе с их соответствующими переводами.

Элемент **label (SHORT_NAME_Type):** Определяет метки сокращенных имен, которые, возможно, были переведены вместе с их соответствующими переводами.

Элемент **label (SYNONYMOUS_NAME_Type):** Определяет метки синонимических имен, которые, возможно, были переведены вместе с их соответствующими переводами.

Элемент **text (TEXT_Type):** Определяет тексты описаний, примечаний и замечаний, которые, возможно, были переведены вместе с их соответствующими переводами.

Определения внутренних типов:

Тип **COUNTRY_CODE_Type:** Является типом лингвистического кода, который представляет собой строку из двух символов и определяет страну в соответствии с ИСО 3166-1.

Тип **DOCUMENT_IDENTIFIER_Type:** Является идентификатором переведенного документа.

Тип **DOCUMENT_IDENTIFIER_NAME_LABEL_Type:** Является строкой (**xs:string** XML-схемы), представляющей собой значения, допускаемые для меток документа вместе с ее языком (XML-атрибут **@language_code** и, возможно, XML-атрибут **@country_code**). Длина этой строки не должна превышать 255 символов.

Тип **GENERAL_TEXT_Type:** Является строкой (**xs:string** XML-схемы), представляющей собой значения, допускаемые для описания, примечания или замечания, вместе с ее языком (XML-атрибут **@language_code** и, возможно, XML-атрибут **@country_code**). Длина этой строки не ограничивается.

Тип **KEYWORD_LABEL_Type:** Является строкой (**xs:string** XML-схемы), представляющей собой значения, допускаемые для ключевого слова вместе с ее языком (XML-атрибут **@language_code** и, возможно, XML-атрибут **@country_code**). Длина этой строки не должна превышать 255 символов.

Тип **KEYWORD_Type:** Является предположительно переведенным ключевым словом понятия.

Тип **LANGUAGE_CODE_Type:** Является типом лингвистического кода, который представляет собой строку из двух или трех символов и определяет язык в соответствии с ИСО 639-1 или ИСО 639-2, соответственно.

Тип **PREFERRED_NAME_LABEL_Type:** Является строкой (**xs:string** XML-схемы), представляющей собой значения, допускаемые для предпочтительного имени вместе с его языком (XML-атрибут **@language_code** и, возможно, XML-атрибут **@country_code**). Длина этой строки не должна превышать 255 символов.

Тип **PREFERRED_NAME_Type:** Является предположительно переведенным предпочтительным именем понятия.

Тип **SHORT_NAME_LABEL_Type:** Является строкой (**xs:string** XML-схемы), представляющей собой значения, допускаемые для сокращенного имени вместе с его языком (XML-атрибут **@language_code** и, возможно, XML-атрибут **@country_code**). Длина этой строки не должна превышать 30 символов.

Тип **SHORT_NAME_Type:** Является предположительно переведенным сокращенным именем понятия.

Тип **SYNONYMOUS_NAME_LABEL_Type:** Является строкой (**xs:string** XML-схемы), представляющей собой значения, допускаемые для синонимического имени вместе с его языком (XML-атрибут **@language_code** и, возможно, XML-атрибут **@country_code**). Длина этой строки не должна превышать 255 символов.

Тип **SYNONYMOUS_NAME_Type:** Является предположительно переведенным синонимическим именем понятия.

Тип **TEXT_Type:** Является предположительно переведенным текстом понятия.

8.1.3 Управление переводом

Может предоставляться информация для управления переводом, проводимым на уровне понятий CIIM-онтологии, для чего в OntoML-языке предназначен комплексный XML-тип данных **TRANSLATION_Type** (см. рисунок 45).

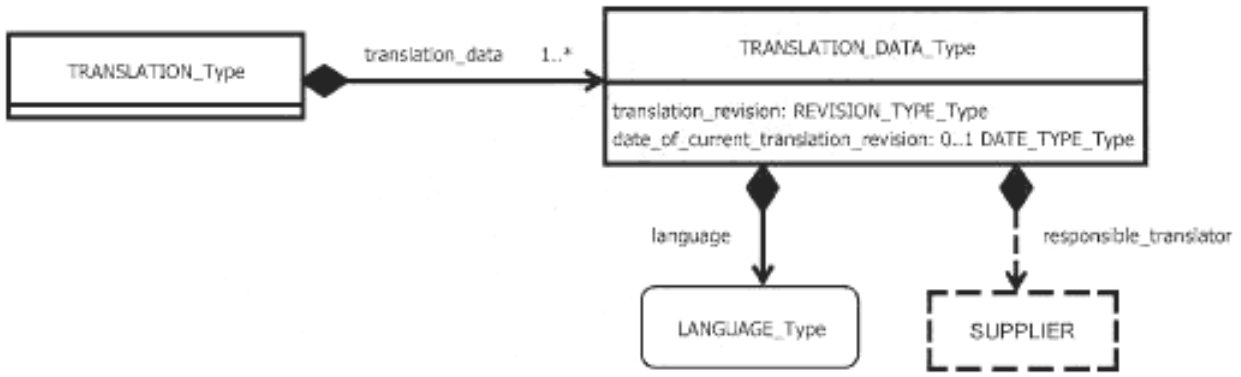


Рисунок 45 — Структура переведенных данных

Определения внутренних элементов:

Элемент **translation_data:** Определяет массив переведенной информации, связанной с понятием CIIM-онтологии.

Элемент **translation_data/date_of_current_translation_revision:** Определяет дату, соответствующую текущей версии перевода на язык с кодом **@language**.

Элемент **translation_data/language:** Определяет язык, на котором приводится переведенная информация.

Элемент **translation_data/responsible_translator:** Определяет ссылку на идентификатор организации, который выполняет перевод на язык с кодом **@language**.

Элемент **translation_data/translation_revision:** Определяет состояние версии перевода на язык с кодом **@language**.

Примечание — Изменение версии или изменение варианта словарного элемента не всегда требует какого-либо изменения в их переводах. При отсутствии изменений в переводе из-за изменения версии или изменения варианта словарного элемента соответствующий элемент **translation_revision** изменяться не должен, однако любые изменения перевода будут подразумевать изменение соответствующего элемента **translation_revision**.

Определения внутренних типов:

Тип **DATE_TYPE_Type:** Является идентификатором значений, допускаемых для обозначения даты (**xs:date** XML-схемы).

Тип **REVISION_TYPE_Type:** Является строкой (**xs:string** XML-схемы), которая представляет значения, допускаемые для обозначения редакции. Эта строка должна содержать не более трех символов.

Тип **TRANSLATION_DATA_Type:** Является переведенной информацией.

Тип **TRANSLATION_Type:** Является информационным контейнером для перевода.

Определение внешнего типа:

Тип **LANGUAGE_Type:** См. 8.1.1.

8.2 Внешний контент

Внешний контент позволяет давать внешние ссылки, определенные из понятий CIIM-онтологии или из информационных элементов. Эти ссылки могут выполняться различными способами, т. е.:

- путем определения унифицированного идентификатора ресурса (URI) — локального или глобального, который будет ссылаться на внешний ресурс;
- путем определения кода, идентифицирующего документ;
- путем определения ссылки на документ или на зарегистрированный в документе графический материал, который описывается или идентифицируется онтологическим понятием документа.

8.2.1 Онтология элементарного уровня для внешних ресурсов

Фрагмент информации может предоставляться в качестве внешнего ресурса, которым может быть:

- либо файл (или набор файлов), связанный с экземпляром OntoML-документа и идентифицированный с помощью локального унифицированного идентификатора ресурса (URI), или
- Интернет-ресурс, идентифицированный с помощью глобального унифицированного идентификатора ресурса.

Рисунок 46 иллюстрирует корневой тип внешнего ресурса, представляемый с помощью абстрактного комплексного XML-типа данных **EXTERNAL_RESOURCE_Type**.

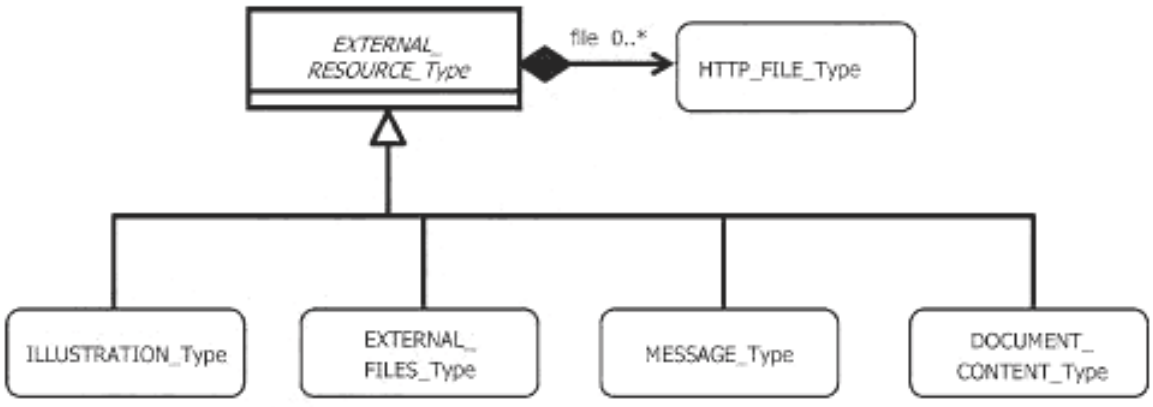


Рисунок 46 — Онтология элементарного уровня для внешних ресурсов

Определение внутреннего элемента:

Элемент **file:** Определяет множество XML-элементов, которые описывают и идентифицируют внешние ресурсы, представляемые с помощью HTTP-файлов.

Определение внутреннего типа:

Тип **EXTERNAL_RESOURCE_Type:** Является абстрактным внешним ресурсом.

Определения внешних типов:

Тип **DOCUMENT_CONTENT_Type:** См. 6.7.7.

Тип **EXTERNAL_FILES_Type:** См. 8.2.1.4.

Тип **HTTP_FILE_Type:** См. 8.2.1.1.

Тип **ILLUSTRATION_Type:** См. 8.2.1.2.

Тип **MESSAGE_Type:** См. 8.2.1.3.

8.2.1.1 HTTP-файл

HTTP-файл является основной OntoML-конструкцией для ссылок на внешнюю информацию из экземпляра OntoML-документа.

HTTP-файл определен в соответствии с комплексным XML-типом данных **HTTP_FILE_Type,** показанным на рисунке 47.

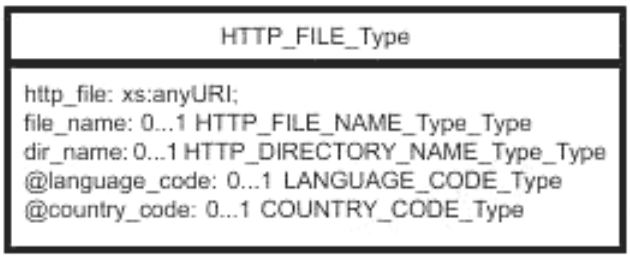


Рисунок 47 — Онтология элементарного уровня для внешних ресурсов: Структура HTTP-файла

Определения внутренних элементов:

Элемент **@country_code:** Определяет допустимый код страны, который в дальнейшем будет определять язык.

Элемент **@language_code:** Определяет допустимый язык, на котором будет выражаться информация, содержащаяся в HTTP-файле.

Элемент **dir_name:** Определяет допустимый целевой каталог, в котором должен сохраняться HTTP-файл в приемной системе, если http-файлы будут включать в себя ссылки друг на друга.

Примечание 1 — Если элемент **dir_name** существует, то все каталоги для одного и того же экземпляра OntoML-документа будут определяться общим корнем.

Элемент **http_file:** Определяет унифицированный идентификатор ресурса (URI), находящийся в HTTP-файле.

Примечание 2 — Интерпретация MIME-протокола полностью определяет протокол, который должен использоваться для обработки любой внешней ссылочной информации.

Элемент **http_file_name:** Определяет допустимое имя HTTP-файла в приемной системе.

Определения внутренних типов:

Тип **COUNTRY_CODE_Type:** Является типом лингвистического кода, представляющего собой строку из двух символов, которые определяют страну в соответствии с ИСО 3166-1.

Тип **HTTP_FILE_NAME_Type**: Является типом элемента **http_file_name,** представление которого отвечает ограничительным условиям, определенным для представляющих их унифицированных идентификаторов ресурса (URI).

Примечание 3 — Унифицированный идентификатор ресурса определен с помощью [RFC 2396] и скорректирован с помощью [RFC 2732].

Тип **HTTP_DIRECTORY_NAME_Type:** Является типом имени каталога для http-файла (элемент dir_name), длина которого не должна превышать 128 символов.

Тип **LANGUAGE_CODE_Type:** Является типом лингвистического кода, представляющего собой строку из двух или трех символов, которые определяют язык в соответствии с ИСО 639-1 или ИСО 639-2, соответственно.

8.2.1.2 Иллюстрация

Иллюстрация представляет собой информативный элемент в виде схематического чертежа, реалистической картинки или любого другого информационного компонента, не являющегося статическим изображением, содержание которого определяется http-файлом (возможно, переведенным). Иллюстрацией также может быть любое изображение в стандартном формате *A6* или *A9*. Если иллюстрация не предоставляется в подобных стандартных форматах, то может определяться рекомендуемый размер «окна» для ее отображения. Иллюстрация представляется комплексным XML-типом **ILLUSTRATION_Type** (см. рисунок 48).

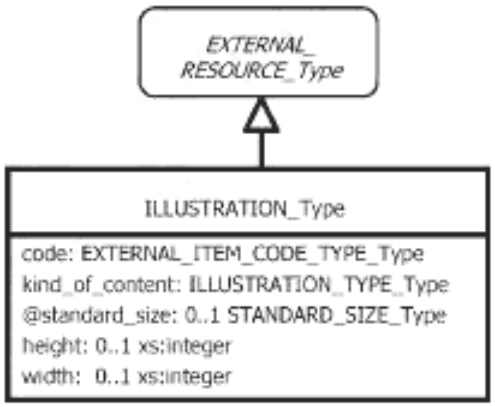


Рисунок 48 — Онтология элементарного уровня для внешних ресурсов: Иллюстрация

Определения внутренних элементов:

Элемент **@standard_size:** Если он предоставляется, то определяет стандартный формат иллюстрации — *A6* или *A9*.

Элемент **code:** Определяет код, идентифицирующий иллюстрацию.

Элемент **height:** В случае не стандартизированного формата иллюстрации определяет высоту «окна», рекомендованную поставщиком данных библиотеки данных для просмотра иллюстрации.

Элемент **kind_of_content:** Определяет категорию содержания иллюстрации — схематического чертежа, реалистической картинки или нестатического изображения.

Элемент **width:** В случае нестандартизированного формата иллюстрации определяет ширину «окна», рекомендованную поставщиком библиотеки данных для просмотра иллюстрации.

Примечание — По умолчанию высота и ширина иллюстрации выражается в миллиметрах.

Определения внутренних типов:

Тип **EXTERNAL_ITEM_CODE_TYPE_Type:** Является элементарным XML-типом данных, определяющим ограничение строки, которая не должна превышать 18 символов и содержать пробелов.

Тип **ILLUSTRATION_TYPE_Type:** Является элементарным XML-типом данных, определяющим ограничение строки, которая должна принадлежать либо "SCHEMATIC_DRAWING", либо "REALISTIC_PICTURE", либо "NOT_STATIC_PICTURE".

Тип **STANDARD_SIZE_Type:** Является элементарным XML-типом данных, определяющим ограничение строки, которая должна принадлежать либо формату "a6_illustration", либо формату "a9_illustration".

Определение внешнего типа:

Тип **EXTERNAL_RESOURCE_Type:** См. 8.2.1.

Перечень ограничительных условий:

Значение элемента **code** является уникальным в том классе, где он используется для определения иллюстрации.

Определяются либо оба элемента **height** и **width**, либо ни один из них.

8.2.1.3 Сообщение

Сообщение — это короткий текст, обычно содержащий не более 255 символов, который по предположению будет автоматически отображаться на экране пользователя библиотеки в четко определенном рабо-

чем контексте. Сообщение не связано с каким бы то ни было конкретным размером «окна», может предоставляться на любом языке и сохраняться в http-файле (для каждого языка — в отдельном файле). Сообщение представляется комплексным XML-типом данных **MESSAGE_Type** (см. рисунок 49).

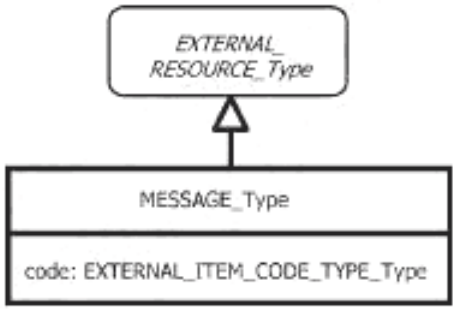


Рисунок 49 — Онтология элементарного уровня для внешних ресурсов: Сообщение

Определение внутреннего элемента:

Элемент **code:** Определяет код, идентифицирующий сообщение.

Определение внутреннего типа:

Тип **EXTERNAL_ITEM_CODE_TYPE_Type:** Является элементарным XML-типом данных, определяющим ограничение строки, которая не должна содержать более 18 символов и не иметь пробелов.

Определение внешнего типа:

Тип **EXTERNAL_RESOURCE_Type:** См. 8.2.1.

Перечень ограничительных условий:

Значение элемента **code** для сообщения является уникальным в том классе, где он используется для определения иллюстрации.

8.2.1.4 Внешние файлы

Внешние файлы предоставляются для обмена графической информацией с помощью ссылок на внешние ресурсы и представляются с помощью комплексного XML-типа данных **EXTERNAL_FILES_Type** (см. рисунок 50).

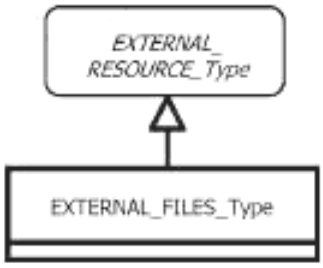


Рисунок 50 — Онтология элементарного уровня для внешних ресурсов: Внешние файлы

Определение внешнего типа:

Тип **EXTERNAL_RESOURCE_Type:** См. 8.2.1.

8.2.2 Документы — источники информации и графические материалы

Документация, которая может быть связана с понятием CIIM-онтологии, представляется в двух вариантах, т. е.:

- документы — источники информации; общий структурный компонент для ссылок на материалы типа документов;

- графические материалы; общий структурный компонент для ссылок на изобразительные материалы.

8.2.2.1 Документы — источники информации

Внешние ресурсы, являющиеся документами—источниками информации, представляются с помощью абстрактного комплексного XML-типа данных **SOURCE_DOCUMENT_Type** (см. рисунок 51).

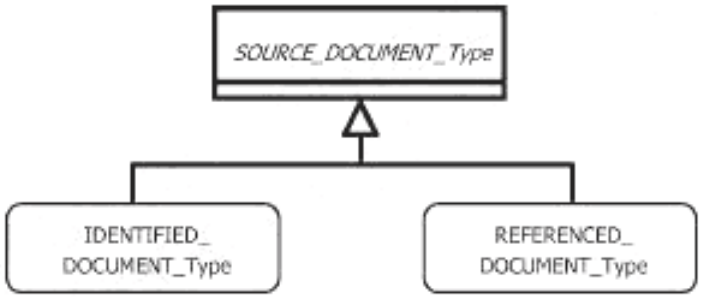


Рисунок 51 — Внешние ресурсы: Документы — источники информации

Определения внешних типов:

Тип **IDENTIFIED_DOCUMENT_Type:** См. 8.2.2.1.1.

Тип **REFERENCED_DOCUMENT_Type:** См. 8.2.2.1.2.

8.2.2.1.1 Идентифицированный документ

Идентифицированный документ — это документ, отождествляемый с помощью собственной метки и представляемый с помощью комплексного XML-типа данных **IDENTIFIED_DOCUMENT_Type** (см. рисунок 52).

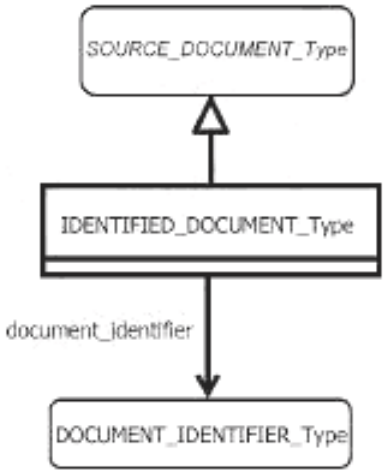


Рисунок 52 — Внешние ресурсы: Идентифицированный документ

Определение внутреннего элемента:

Элемент **document_identifier:** Определяет метку (возможно, переведенную) описываемого документа.

Определения внешних типов:

Тип **DOCUMENT_IDENTIFIER_Type:** См. 8.1.2.

Тип **SOURCE_DOCUMENT_Type:** См. 8.2.2.1.

8.2.2.1.2 Ссылочный документ

Ссылочный документ позволяет давать ссылку на документ, который описывается и идентифицируется с помощью онтологического понятия документа, а также представляется с помощью комплексного XML-типа данных **REFERENCED_DOCUMENT_Type** (см. рисунок 53).

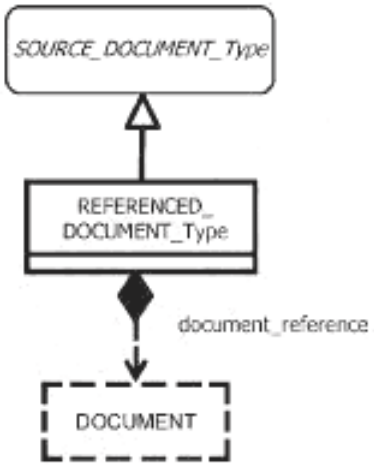


Рисунок 53 — Внешние ресурсы: Ссылочный документ

Определение внутреннего элемента:

Элемент **document_reference:** Определяет ссылку на идентификатор онтологического понятия документа.

Определение внешнего типа:

Тип **SOURCE_DOCUMENT_Type:** См. 8.2.2.1.

8.2.2.2 Графические материалы

Внешние ресурсы, предоставляемые в виде графических материалов, представляются в виде абстрактного комплексного XML-типа данных **GRAPHICS_Type** (см. рисунок 54).

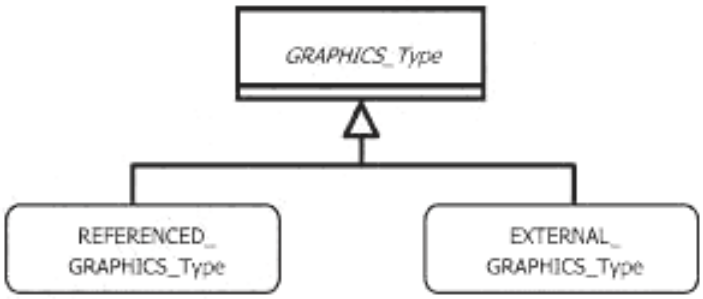


Рисунок 54 — Внешние ресурсы: Графические материалы

Определения внешних типов:

Тип **EXTERNAL_GRAPHICS_Type:** См. 8.2.2.2.1.

Тип **REFERENCED_GRAPHICS_Type:** См. 8.2.2.2.2.

8.2.2.2.1 Внешние графические материалы

Внешние графические материалы позволяют описывать графику и представляются в виде комплексного XML-типа данных **EXTERNAL_GRAPHICS_Type** (см. рисунок 55).

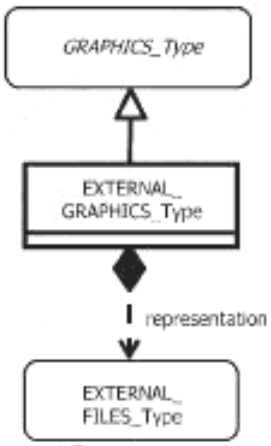


Рисунок 55 — Внешние ресурсы: Внешние графические материалы

Определение внутреннего элемента:

Элемент **representation:** Определяет внешний файл (файлы), определяющий внешние графические материалы.

Определения внешних типов:

Тип **EXTERNAL_FILES_Type:** См. 8.2.1.4.

Тип **GRAPHICS_Type:** См. 8.2.2.2.

Перечень ограничительных условий:

Тип **EXTERNAL_GRAPHICS_Type** должен предоставлять значение в наследуемый XML-элемент данных **file**.

8.2.2.2.2 Ссылочные графические материалы

Ссылочные графические материалы позволяют давать ссылку на графику, которая описывается и идентифицируется с помощью онтологического понятия документа, а также представляется с помощью комплексного XML-типа данных **REFERENCED_GRAPHICS_Type** (см. рисунок 56).

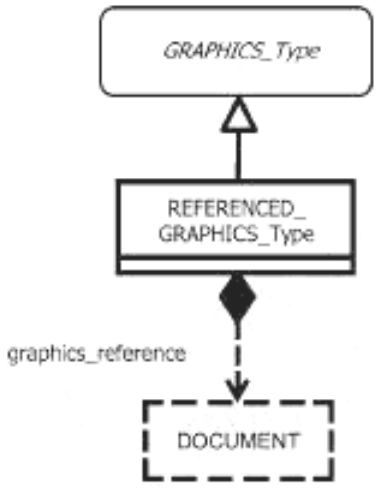


Рисунок 56 — Внешние ресурсы: Ссылочные графические материалы

Определение внутреннего элемента:

Элемент **graphics_reference:** Определяет ссылку на идентификатор онтологического понятия документа.

Определение внешнего типа:

Тип **GRAPHICS_Type:** См. 8.2.2.2.

Перечень ограничительных условий:

Тип **REFERENCED_GRAPHICS_Type** не должен предоставлять значение в наследуемый XML-элемент данных **file**.

8.3 Система типов данных

OntoML-язык предоставляет ресурсы для описания типов данных, что позволяет ограничивать область значений, закрепленных за определенным свойством (признаком). При этом доступными являются как простые типы данных (булев, действительный, строковый и т. п.), так и сложные типы данных (именованные, сборные, классные и т. п.). Каждый тип данных определяется как подтип комплексного XML-типа данных **ANY_TYPE_Type**. Рисунок 57 иллюстрирует основные типы данных, доступные в OntoML-языке.

Примечание 1 — Лексическое представление значения каждого типа данных, принадлежащего OntoML-системе типов данных, дано в приложении D. В следующем подразделе каждый тип данных будет сопоставляться с соответствующим лексическим представлением.

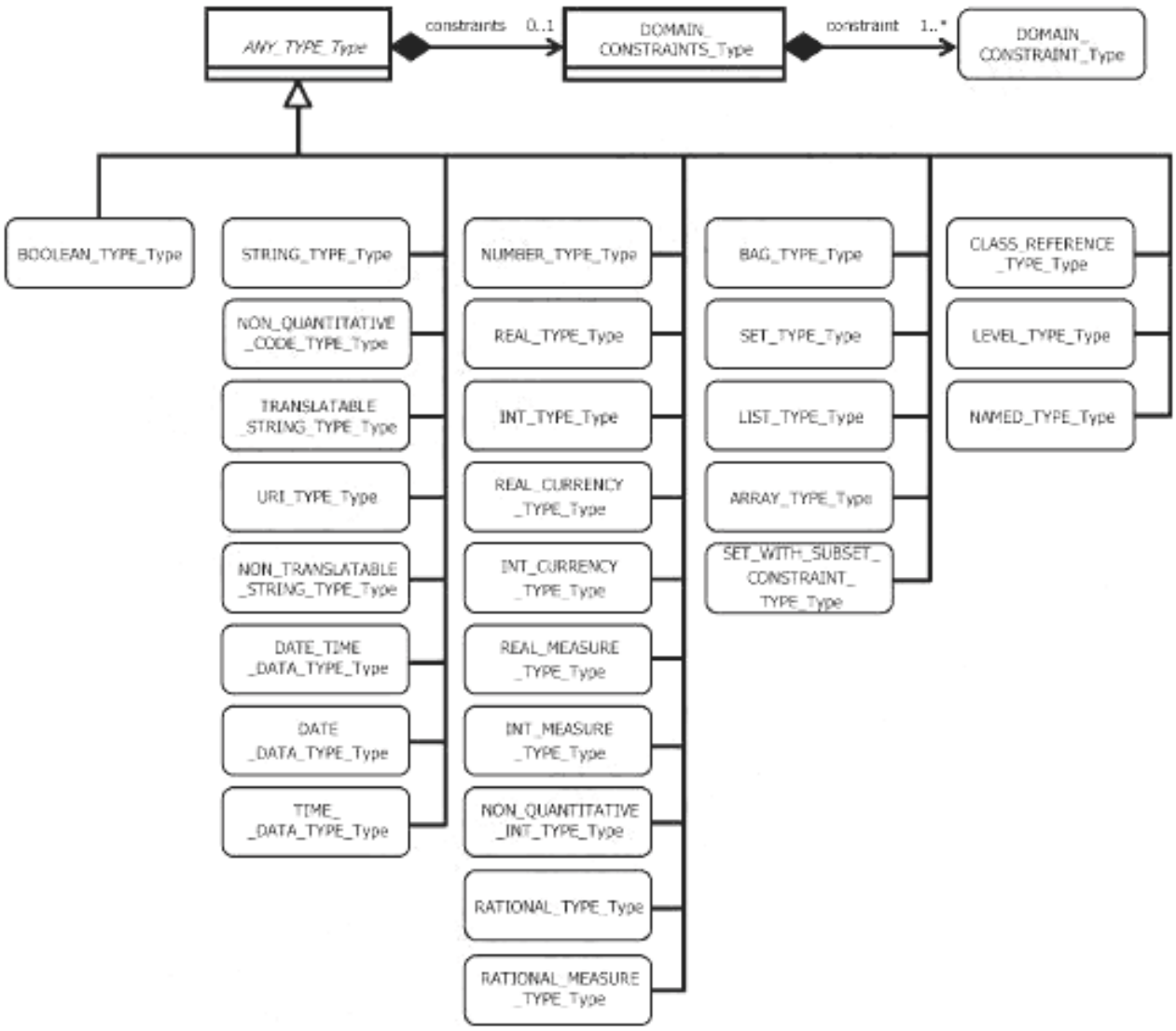


Рисунок 57 — OntoML-система типов данных

Определение внутреннего элемента:

Элемент **constraints:** Определяет совокупность условий для ограничения области значений типа данных.

Примечание 2 — Каждое ограничение области значений должно полностью выполняться, поэтому совокупность ограничений будет определять их связи.

Определение внутреннего типа:

Тип **DOMAIN_CONSTRAINTS_Type:** Является группой ограничительных условий для области значений.

Определения внешних типов:

Тип **ARRAY_TYPE_Type:** Является группой массивов, см. 8.3.9.4.

Тип **BAG_TYPE_Type:** Является группой пакетов, см. 8.3.9.1.

Тип **BOOLEAN_TYPE_Type:** Является булевым типом данных, см. 8.3.1.

Тип **CLASS_REFERENCE_TYPE_Type:** Является ссылкой на идентифицированный тип класса, см. 8.3.10.

Тип **DATE_TIME_DATA_TYPE_Type:** Является типом данных «дата/время», см. 8.3.3.

Тип **DATE_DATA_TYPE_Type:** Является типом данных «дата», см. 8.3.3.

Тип **DOMAIN_CONSTRAINT_Type:** Является типом данных «ограничительные условия», см. 8.5.3.3.

Тип **INT_CURRENCY_TYPE_Type:** Является типом данных «целое значение валюты», см. 8.3.6.

Тип **INT_MEASURE_TYPE_Type:** Является типом целого значения без единицы измерений, см. 8.3.7.
Тип **INT_TYPE_Type:** Является типом целого числа без единицы измерения, см. 8.3.5.
Тип **LEVEL_TYPE_Type:** Является типом уровня, см. 8.3.11.
Тип **LIST_TYPE_Type:** Является группой списков, см. 8.3.9.3.
Тип **NAMED_TYPE_Type:** Является ссылочным типом на идентифицированные данные, см. 8.3.12.
Тип **NON_QUANTITATIVE_CODE_TYPE_Type:** Является типом перечня строковых кодов, см. 8.3.4.
Тип **NON_QUANTITATIVE_INT_TYPE_Type:** Является типом перечня целочисленных кодов, см. 8.3.8.
Тип **NON_TRANSLATABLE_STRING_TYPE_Type:** Является типом непереводимой строки, см. 8.3.2.
Тип **NUMBER_TYPE_Type:** Является типом номера, см. 8.3.5.
Тип **RATIONAL_TYPE_Type:** Является типом рационального значения, см. 8.3.5.
Тип **RATIONAL_MEASURE_TYPE_Type:** Является типом рациональной меры, см. 8.3.7.
Тип **REAL_TYPE_Type:** Является типом действительного числа без единицы измерений, см. 8.3.5.
Тип **REAL_CURRENCY_TYPE_Type:** Является типом действительного значения валюты, см. 8.3.6.
Тип **REAL_MEASURE_TYPE_Type:** Является типом действительного значения с единицей измерения, см. 8.3.7.
Тип **URI_TYPE_Type:** Является строкой, представляемой URI-идентификатором, см. 8.3.2.
Тип **SET_TYPE_Type:** Является типом установленной группы элементов, см. 8.3.9.2.
Тип **SET_WITH_SUBSET_CONSTRAINT_TYPE_Type:** Является типом четко определенной группы элементов, см. 8.3.9.
Тип **STRING_TYPE_Type:** Является типом строки, см. 8.3.2.
Тип **TIME_DATA_TYPE_Type:** Является типом данных «время», см. 8.3.3.
Тип **TRANSLATABLE_STRING_TYPE_Type:** Является типом переводимой строки, см. 8.3.2.

8.3.1 Булев тип данных

Булев тип данных определяется с помощью комплексного XML-типа данных **BOOLEAN_TYPE_Type** (см. рисунок 58).

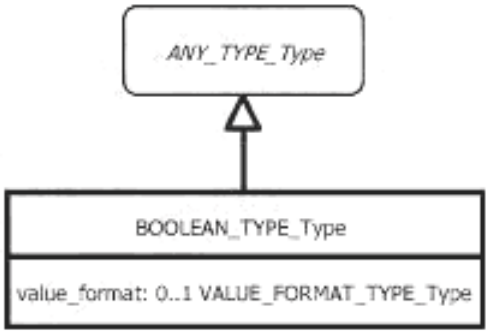


Рисунок 58 — Структура булева типа данных

Определение внутреннего элемента:

Элемент **value_format:** Определяет тип и длину рекомендованного представления данных для отображения значения свойства. Данный атрибут содержит описание способа отображения этого значения в системе.

Определение внутреннего типа:

Тип **VALUE_FORMAT_TYPE_Type:** Является идентификатором значений, допустимых для использования в данном формате.

Примечание 1 — Для данного элемента не определено стандартное значение.

Перечень ограничительных условий:

Тип **BOOLEAN_TYPE_Type:** Предназначен для значений свойств или типов пользователей, которые принадлежат булеву типу.

Примечание 2 — Лексическое представление значения, чьим типом данных является тип **BOOLEAN_TYPE_type**, приведено в D.1.1 приложения D.

Определение внешнего типа:

Тип **ANY_TYPE_Type:** См. 8.3.

Перечень ограничительных условий:

Длина значения для типа **VALUE_FORMAT_TYPE_Type** не должна превышать 80 символов.

8.3.2 Строковые типы данных

Область значений, определенная с помощью строкового типа данных, является последовательностью произвольных символов. Основной строковый OntoML-тип определяется с помощью комплексного XML-типа **STRING_TYPE_Typ,** который в дальнейшем может классифицироваться как локализованная строка (комплексный XML-тип данных **TRANSLATABLE_STRING_TYPE_Type**), как непереводимая строка (комплексный XML-тип данных **NON_TRANSLATABLE_STRING_TYPE_Type**) или как строка, представляемая HTTP-адресом (комплексный XML-тип данных **URI_TYPE_Type**). Рисунок 59 иллюстрирует указанные ресурсы.

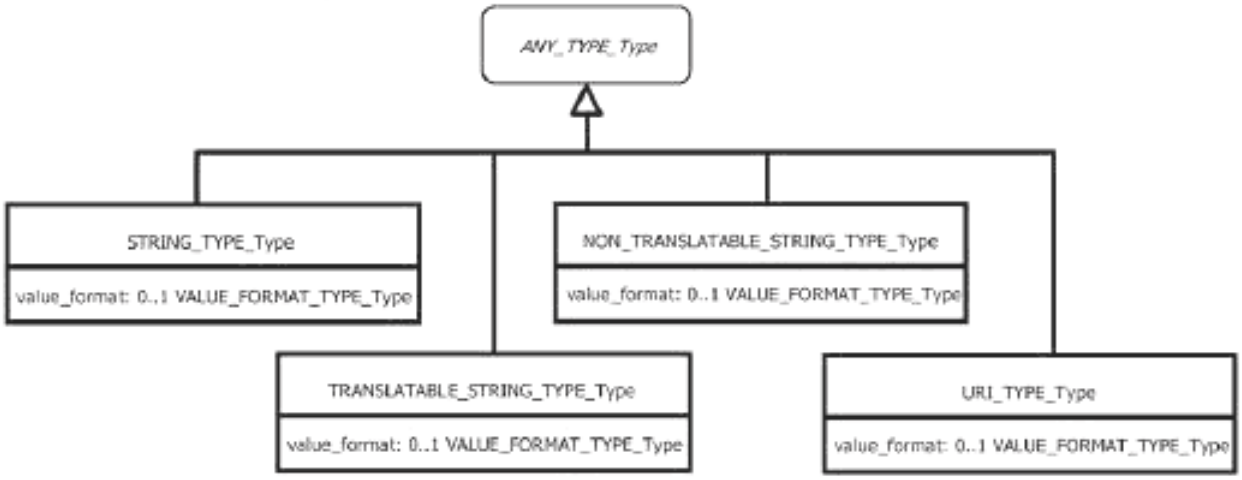


Рисунок 59 — Структура строковых типов данных

Определение внутреннего элемента:

Элемент **value_format** (для типов данных **STRING_TYPE_Type, TRANSLATABLE_STRING_TYPE_Type, NON_TRANSLATABLE_STRING_TYPE_Type** и **URI_TYPE_Type**): Определяет тип и длину рекомендуемого представления для отображения значения свойства. При наличии этого элемента его атрибут обеспечивает введение в систему данных в отношении способа отображения этого значения.

Примечание 1 — Элемент **value_format** не должен использоваться для ограничения определения строкового типа данных.

Примечание 2 — Если элемент **value_format** несовместим со связанным с ним определением строкового типа данных, то этот элемент будет игнорироваться.

Примечание 3 — Если любое ограничительное условие на структуру строки (см. раздел 8.5.3.3.2) будет применимо к значению строкового типа, то оно будет иметь преимущественную силу по отношению к элементу **value_format**.

Определение внутреннего типа:

Тип **VALUE_FORMAT_TYPE_Type:** Является идентификатором значений, допустимых для использования в данном формате.

Примечание 4 — Значения для типа данных **VALUE_FORMAT_TYPE_Type** определены в приложении H.

Перечень ограничительных условий:

Тип **NON_TRANSLATABLE_STRING_TYPE_Type:** Предназначен для значений свойств или типов пользователей, которые принадлежат строковому типу данных, однако представляются одним и тем же способом на любом языке.

Примечание 5 — Лексическое представление значения, чьим типом данных является тип данных **NON_TRANSLATABLE_STRING_TYPE_Type,** приведено в D.1.6 приложения D.

Тип **STRING_TYPE_Type:** Предназначен для значений свойств или типов пользователей, которые принадлежат строковому типу данных.

Примечание 6 — Лексическое представление значения, чьим типом данных является тип данных **STRING_TYPE_Type,** приведено в D.1.2 приложения D.

Тип **TRANSLATABLE_STRING_TYPE_Type:** Предназначен для значений свойств или типов пользователей, которые принадлежат строковому типу данных, однако, как предполагается, на различных языках должны представляться в виде различных строк.

Примечание 7 — Лексическое представление значения, чьим типом данных является тип данных **TRANSLATABLE_STRING_TYPE_Type,** приведено в D.1.4 приложения D.

Примечание 8 — XML-элемент **source_language** (определенный в комплексном XML-типе данных **PROPERTY_DET,** см. 6.7.4) для свойства, чей тип данных (XML-элемент области) определен как тип данных **TRANSLATABLE_STRING_TYPE_Type,** играет роль корневого языка, в котором строковые значения свойства преобразуются с целью идентификации одних и тех же значений при их представлении на различных языках.

Тип **URI_TYPE_Type:** Предназначен для значений свойств или типов пользователей, которые принадлежат строковому типу данных, однако представляют собой унифицированный идентификатор ресурса (URI).

Примечание 9 — Лексическое представление значения, чьим типом данных является тип **URL_TYPE_Type,** приведено в D.1.5 приложения D.

Определение внешнего типа:

Тип **ANY_TYPE_Type:** См. 8.3.

Перечень ограничительных условий:

Значение для типа данных **VALUE_FORMAT_TYPE_Type** не должно превышать 80 символов.

8.3.3 Тип данных «дата/время»

Область значений, определенная с помощью типа данных «дата/время», является последовательностью символов, отвечающих требованиям к представлению данных, которые определены в ИСО 8601. Этот тип может представлять либо и дату и время (комплексный XML-тип данных **DATE_TIME_DATA_TYPE_Type**), либо только дату (комплексный XML-тип данных **DATE_DATA_TYPE_Type**), либо только время (комплексный XML-тип данных **TIME_DATA_TYPE_Type**). Рисунок 60 иллюстрирует указанные ресурсы.

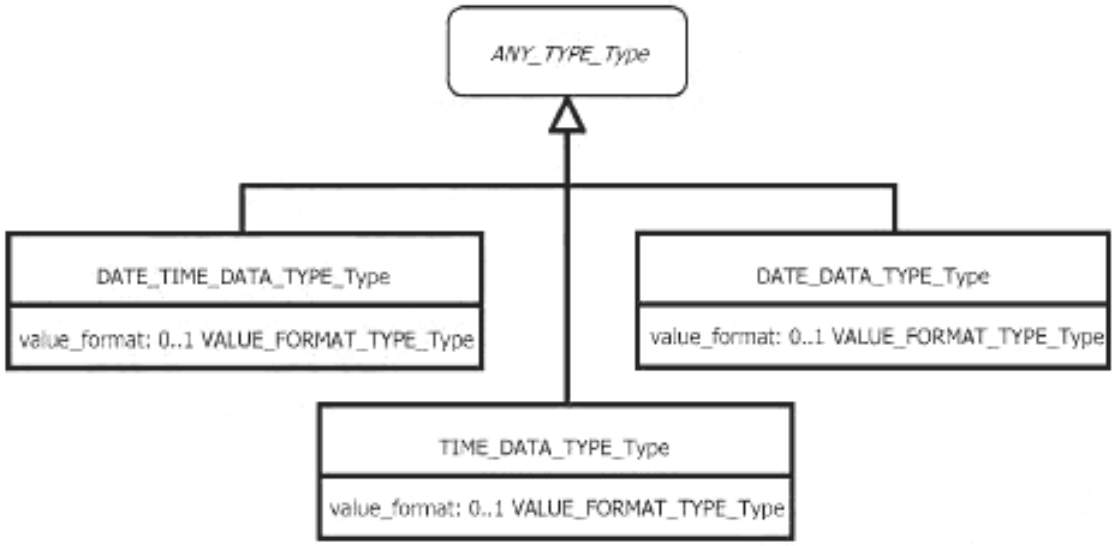


Рисунок 60 — Структура типов даты и времени

Определение внутреннего элемента:

Элемент **value_format** (для типов данных **DATE_TIME_DATA_TYPE_Type, TIME_DATA_TYPE_Type** и **DATE_DATA_TYPE_Type**): Определяет тип и длину рекомендованного представления данных для отображения значения свойства. Данный атрибут содержит системные требования к способу отображения значений.

Примечание 1 — Элемент **value_format** не должен использоваться для ограничения определения типа данных «дата/время».

Примечание 2 — Если элемент **value_format** несовместим со связанным с ним определением типа данных «дата/время», то этот элемент будет игнорироваться.

Примечание 3 — Если любое ограничительное условие на структуру строки (см. раздел 8.5.3.3.2) будет применимо к значению типа данных «дата/время», то оно будет иметь преимущественную силу по отношению к элементу **value_format**.

Определение внутреннего типа:

Тип **VALUE_FORMAT_TYPE_Type:** Является идентификатором значений, допустимых для использования в данном формате.

Примечание 4 — Значения для типа данных **VALUE_FORMAT_TYPE_Type** определены в приложении H.

Перечень ограничительных условий:

Тип **DATE_DATA_TYPE_Type:** Предназначен для значений свойств или типов пользователей, которые принадлежат типу данных «дата».

Примечание 5 — Лексическое представление значения, чьим типом данных является тип **DATE_DATA_TYPE_Type,** приведено в D.1.8 приложения D.

Тип **DATE_TIME_DATA_TYPE_Type:** Предназначен для значений свойств или типов пользователей, которые принадлежат типу данных «дата».

Примечание 6 — Лексическое представление значения, чьим типом данных является тип **DATE_TIME_DATA_TYPE_Type,** приведено в D.1.7 приложения D.

Тип **TIME_DATA_TYPE_Type:** Предназначен для значений свойств или типов пользователей, которые принадлежат типу данных «время».

Примечание 7 — Лексическое представление значения, чьим типом данных является тип **TIME_DATA_TYPE_Type,** приведено в D.1.9 приложения D.

Определение внешнего типа:

Тип **ANY_TYPE_Type:** См. раздел 8.3.

Перечень ограничительных условий:

Значение для типа данных **VALUE_FORMAT_TYPE_Type** не должно превышать 80 символов.

8.3.4 Перечень типов строковых кодов

Строка, которая должна брать свои значения из множества перечислимых кодов, представляется с помощью комплексного XML-типа данных **NON_QUANTITATIVE_CODE_TYPE_Type,** причем каждый из этих кодов связывается со своим содержанием (см. рисунок 61).

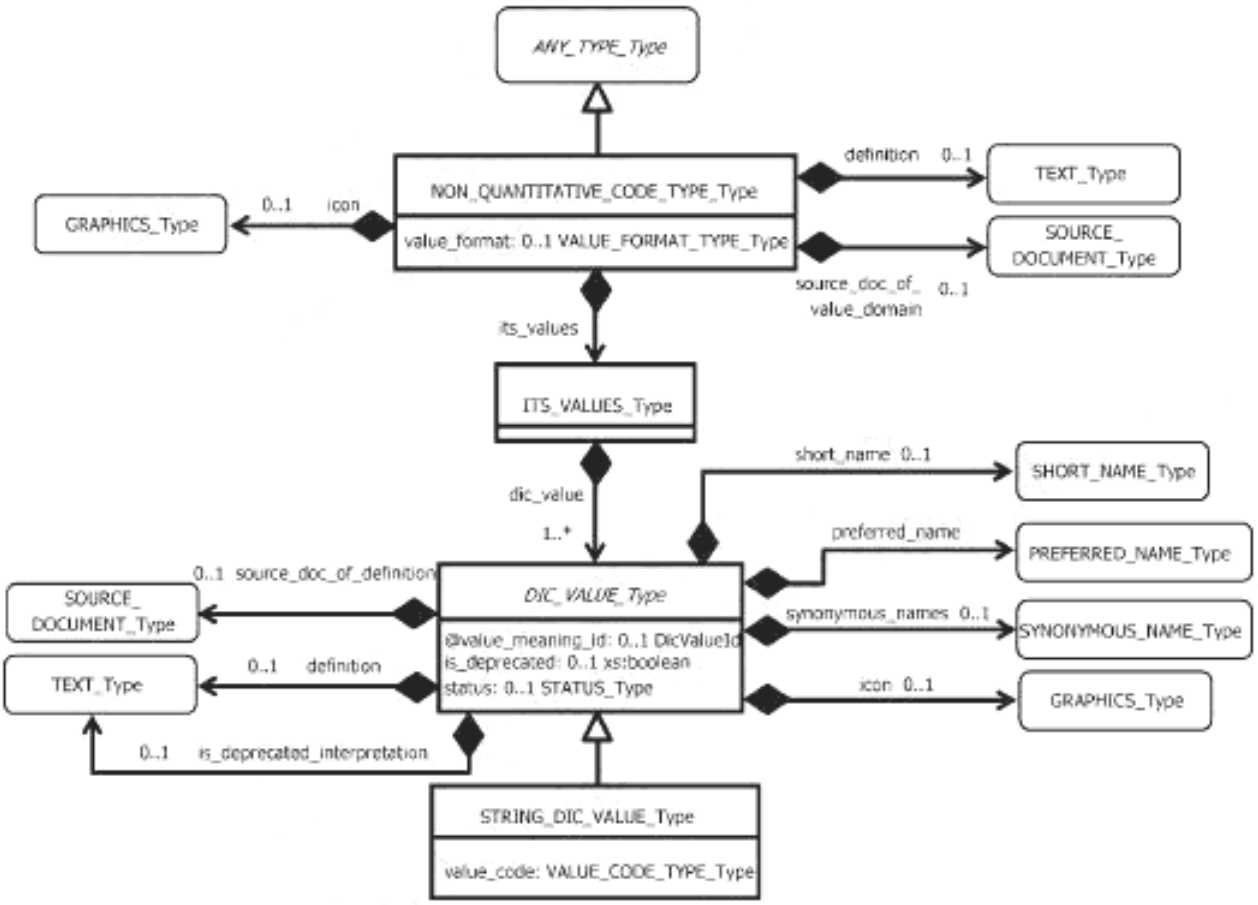


Рисунок 61 — Структура типа перечислимых строковых кодов

Перечислимые элементы определяются в хранилище XML-элемента, называемом **its_values,** чья модель содержания определяется с помощью комплексного XML-типа данных **ITS_VALUES_Type**. Каждый из перечислимых элементов представляется посредством XML-элемента **dic_value**. Его собственная модель содержания определяется типом данных **DIC_VALUE_Type,** или, более точно, — в случае спецификации на перечисление строковых кодов — с помощью собственного специального подтипа комплексного XML-типа данных **STRING_DIC_VALUE_Type**.

Определения внутренних элементов:

Элемент **definition:** Определяет текст, описывающий данный перечень (возможно, переведенный).

Элемент **icon:** Определяет графические материалы, представляющие описание, которое связано с перечнем типов.

Элемент **its_values:** Определяет хранилище элементов перечня.

Элемент **its_values/dic_value:** Определяет элемент перечня.

Примечание 1 — Каждый XML-элемент **dic_value** модели содержания представляется как значения **Instance** комплексного XML-типа данных **STRING_DIC_VALUE_Type**.

Элемент **its_values/dic_value/definition:** Определяет текст, описывающий этот элемент перечня (возможно, переведенного).

Элемент **its_values/dic_value/icon:** Определяет графические материалы, представляющие описание, которое связано с перечнем имен типов.

Элемент **its_values/dic_value/is_deprecated:** Определяет булево значение перечня (если оно истинно), которое не должно больше использоваться.

Примечание 2 — Если элемент **is_deprecated** не определен, то элемент **dic_value** не исключается.

Примечание 3 — Исключенный элемент **dic_values** остается в группе элементов **its_values** по вышеприведенным соображениям совместимости.

Элемент **its_values/dic_value/is_deprecated_interpretation:** Определяет обоснование исключения и способ интерпретации значений экземпляров исключаемого элемента.

Примечание 4 — Значения экземпляров исключаемого элемента должны определяться во время принятия решения об их исключении.

Элемент **its_values/dic_value/preferred_name:** Определяет имя элемента перечня, которое предпочтительно для использования (возможно, переведенное).

Элемент **its_values/dic_value/short_name:** Определяет сокращение предпочтительного имени (возможно, переведенное).

Элемент **its_values/dic_value/source_doc_of_definition:** Определяет возможный документ-источник, из которого исходит определение элемента списка.

Элемент **its_values/dic_value/status:** Определяет состояние элемента перечня в его жизненном цикле.

Примечание 5 — Допустимые значения **status** определяются путем заключения частного соглашения между поставщиком словаря и его пользователями.

Примечание 6 — Если XML-элемент **status** не предоставляется и если элемент перечня не исключен как обозначенный с помощью возможного XML-элемента **is_deprecated,** то элемент перечня имеет то же состояние стандартизации, что и онтология, в которой он используется. В частности, если онтология стандартизована, ее элемент перечня будет являться частью текущей редакции стандарта.

Элемент **its_values/dic_value/synonymous_names:** Определяет набор синонимических имен предпочтительного имени (возможно, переведенных).

Элемент **its_values/dic_value/value_code:** Определяет значение элемента перечня.

Элемент **its_values/dic_value/@value_meaning_id:** Определяет идентификатор, который является глобальным идентификатором значения, независимо от области значений, в которую он включен.

Примечание 7 — Данный идентификатор позволяет повторно использовать одно и то же определение элемента **dic_value** в различных областях значений. Элемент **source_doc_of_value_domain:** возможный документ-источник, из которого исходит определение перечня.

Элемент **value_format:** Определяет описание типа и длины рекомендуемого представления для отображения значения свойства. Данный атрибут (при его наличии) содержит указание относительно способа отображения значения в системе.

Примечание 8 — Элемент **value_format** не должен использоваться для ограничения перечня определения кода строкового типа.

Примечание 9 — Если элемент **value_format** несовместим со связанным с ним определением перечня кодов строкового типа, то значение **value_format** будет игнорироваться.

Примечание 10 — Если любое ограничительное условие на характеристику строки (см. 8.5.3.3.2) применимо к значению перечня строкового типа кодов, то оно будет иметь преимущественную силу по отношению к элементу **value_format**.

Определения внутренних типов:

Тип **STATUS_Type:** Является идентификатором строки (**xs:string** XML-схемы), который представляет значения, допустимые для данного состояния.

Тип **VALUE_FORMAT_TYPE_Type:** Определяет значения, допустимые для использования в данном формате.

Примечание 11 — Значения типа данных **VALUE_FORMAT_TYPE_Type** определены в соответствии с приложением H.

Перечень ограничительных условий:

Тип **NON_QUANTITATIVE_CODE_TYPE_Type:** Приводит значения свойств или типов данных, которые являются перечнем кодов строкового типа.

Примечание 12 — Лексическое представление значения, чьим типом данных является тип **NON_QUANTITATIVE_CODE_TYPE_Type**, приведено в разделе D.1.3 приложения D.

Определение внешних типов:

Тип **ANY_TYPE_Type:** См. 8.3.

Тип **GRAPHICS_Type:** См. 8.2.2.2.

Тип **PREFERRED_NAME_Type:** См. 8.1.2.

Тип **SHORT_NAME_Type:** См. 8.1.2.

Тип **SOURCE_DOCUMENT_Type:** См. 8.2.2.1.

Тип **SYNONYMOUS_NAME_Type:** См. 8.1.2.

Тип **TEXT_Type:** См. 8.1.2.

Перечень ограничительных условий:

Примерные значения элемента **is_deprecated_interpretation** должны определяться в тот момент, когда принималось решение об исключении.

Значение типа данных **VALUE_FORMAT_TYPE_Type** не должно содержать более 80 символов.

8.3.5 Типы численных данных

Численное значение величины может представляться в виде общего значения (типа данных **NUMBER_TYPE_Type**) или реального значения (типа данных **REAL_TYPE_Type**), или в виде целого числа (типа данных **INT_TYPE_Type**), или рационального числа (типа данных **RATIONAL_TYPE_Type**).

Представление численных данных приведено на рисунке 62.

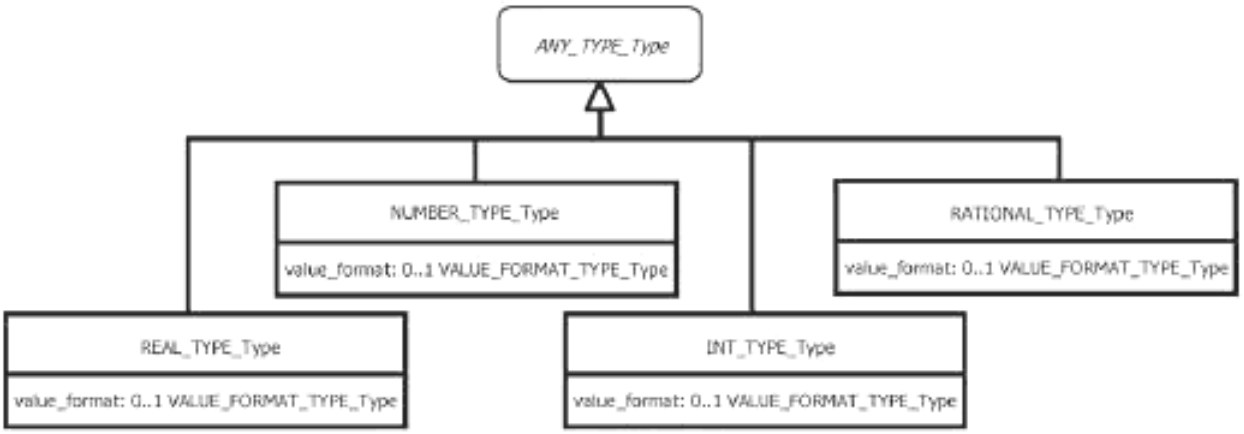


Рисунок 62 — Структура типов численных данных

Определение внутреннего элемента:

Элемент **value_format** (типы данных **NUMBER_TYPE_Type, REAL_TYPE_Type, INT_TYPE_Type, RATIONAL_TYPE_Type**): Определяет описание типа и длины рекомендуемого представления для отображения значения свойства. Данный атрибут (при его наличии) содержит указание относительно способа отображения значения в системе.

Примечание 1 — Элемент **value_format** не должен использоваться для ограничения определения численных данных.

Примечание 2 — Если элемент **value_format** несовместим со связанным с ним определением численных данных, то элемент **value_format** будет игнорироваться.

Примечание 3 — Если любое ограничительное условие на характеристику строки (см. раздел 8.5.3.3.2) применимо к значению численного типа, то оно будет иметь преимущественную силу по сравнению с элементом **value_format**.

Определение внутреннего типа:

Тип **VALUE_FORMAT_TYPE_Type:** Является идентификатором значений, допустимых для использования в данном формате.

Примечание 4 — Значения для типа данных **VALUE_FORMAT_TYPE_Type** определены в соответствии с приложением H.

Перечень ограничительных условий:

Тип **INT_TYPE_Type:** Определяет значения свойств или типов пользователя, которые принадлежат типу целых чисел (integer).

Примечание 5 — Лексическое представление значения, чьим типом данных является тип **INT_TYPE_Type,** приведено в D.1.12 приложения D.

Тип **NUMBER_TYPE_Type**: Определяет значения свойств или типов пользователя, принадлежащих к числовому типу (number).

Примечание 6 — Лексическое представление значения, чьим типом данных является тип **NUMBER_TYPE_Type,** приведено в D.1.10 приложения D.

Тип **RATIONAL_TYPE_Type:** Определяет значения свойств или типов пользователя, которые принадлежат типу рациональных чисел (rational).

Примечание 7 — Лексическое представление значения, чьим типом данных является тип **RATIONAL_TYPE_Type,** приведено в D.1.10 приложения D.

Тип **REAL_TYPE_Type:** Определяет значения свойств или типов пользователя, которые принадлежат типу действительных чисел (real).

Примечание 8 — Лексическое представление значения, чьим типом данных является тип **REAL_TYPE_Type,** приведено в D.1.11 приложения D.

Определение внешнего типа:

Тип **ANY_TYPE_Type:** См. 8.3.

Перечень ограничительных условий:

Значение типа данных **VALUE_FORMAT_TYPE_Type** не должно содержать более 80 символов.

8.3.6 Числовые типы данных «валюта»

Валюта (currency) может выражаться с помощью областей целых и действительных значений. Первое из этих значений представляется с помощью OntoML-комплексного XML-типа данных **INT_CURRENCY_TYPE_Type,** а второе — с помощью OntoML-комплексного XML-типа данных **REAL_CURRENCY_TYPE_Type**. Рисунок 63 иллюстрирует представление типов числовых значений «валюта».

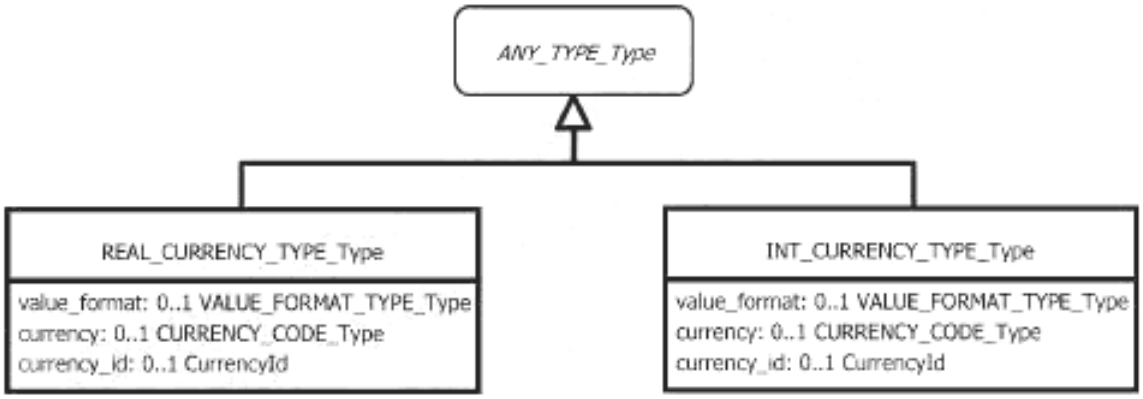


Рисунок 63 — Структура типов числовых значений «валюта»

Определения внутренних элементов:

Элемент **currency** (типы данных **REAL_CURRENCY_TYPE_Type, INT_CURRENCY_TYPE_Type**): Определяет возможный код, связанный с рассматриваемой валютой.

Примечание 1 — Валюта выражается согласно ИСО 4217.

Примечание 2 — Если валюта не определена, то она предназначена для явного определения на уровне библиотеки.

Элемент **currency_id** (типы данных **REAL_CURRENCY_TYPE_Type, INT_CURRENCY_TYPE_Type**): Определяет возможный идентификатор, связанный с описываемой валютой.

Примечание 3 — Если предоставляются элементы **currency** и **currency_id,** то первый элемент будет иметь преимущественную силу.

Примечание 4 — Если значение свойства, чьей областью значений является валюта, обменивается как одиночное число, то считается, что это значение выражается с помощью элемента **currency** или **currency_id** валюты.

Элемент **value_format** (типы данных **REAL_CURRENCY_Type, INT_CURRENCY_Type**): Определяет тип и длину рекомендуемого представления для отображения значения свойства. Данный атрибут (при его наличии) содержит указание относительно способа отображения значения в системе.

Примечание 5 — Элемент **value_format** не должен использоваться для определения типа валюты.

Примечание 6 — Если элемент **value_format** несовместим со связанным с ним определением типа валюты, то значение **value_format** будет игнорироваться.

Примечание 7 — Если любое ограничительное условие на характеристику строки (см. раздел 8.5.3.3.2) применимо к значению типа валюты, то оно будет иметь преимущественную силу по отношению к элементу **value_format**.

Определения внутренних типов:

Тип **CURRENCY_CODE_Type:** Является строкой (**xs:string** XML-схемы), которая представляет значения, допускаемые для валюты. Строка должна содержать три символа.

Пример — Значение валюты может быть «CHF» для французских франков, «CNY» — для юаней КНР, «JPY» — для японских йен, «SUR» — для рублей РФ, «USD» — для долларов США, «EUR» — для евро CE и т. п.

Элемент **CurrencyId:** См. 9.1.4.

Тип **VALUE_FORMAT_TYPE_Type:** Является идентификатором значений, допустимых для использования в данном формате.

Примечание 8 — Значения для типа данных **VALUE_FORMAT_TYPE_Type** определены в соответствии с приложением H.

Перечень ограничительных условий:

Тип **INT_CURRENCY_TYPE_Type:** Приводит значения свойств или типов данных, которые являются целочисленным типом валюты.

Примечание 9 — Лексическое представление значения, чьим типом данных является тип **INT_CURRENCY_TYPE_Type,** приведено в D.1.15 приложения D.

Тип **REAL_CURRENCY_TYPE_Type:** Определяет значения свойств или типов данных, которые являются действительным типом валюты.

Примечание 10 — Лексическое представление значения, чьим типом данных является тип **REAL_CURRENCY_TYPE_Type,** приведено в D.1.16 приложения D.

Определение внешнего типа:

Тип **ANY_TYPE_Type**: См. 8.3.

Перечень ограничительных условий:

Предоставляется либо элемент **currency,** либо элемент **currency_id,** либо оба элемента.

Значение для типа данных **VALUE_FORMAT_TYPE_Type** не должно превышать 80 символов.

8.3.7 Типы данных «численная мера»

Области целочисленных (integer), действительных (real) и рациональных (rational) значений могут представлять меру. Первое из значений представляется с помощью OntoML-комплексного XML-типа данных **INT_MEASURE_TYPE_Type,** второе — с помощью OntoML-комплексного XML-типа данных **REAL_MEASURE_TYPE_Type,** а третье — с помощью комплексного XML-типа данных **RATIONAL_MEASURE_TYPE_Type**. Рисунок 64 иллюстрирует представление типов данных «численная мера (numeric measure)».

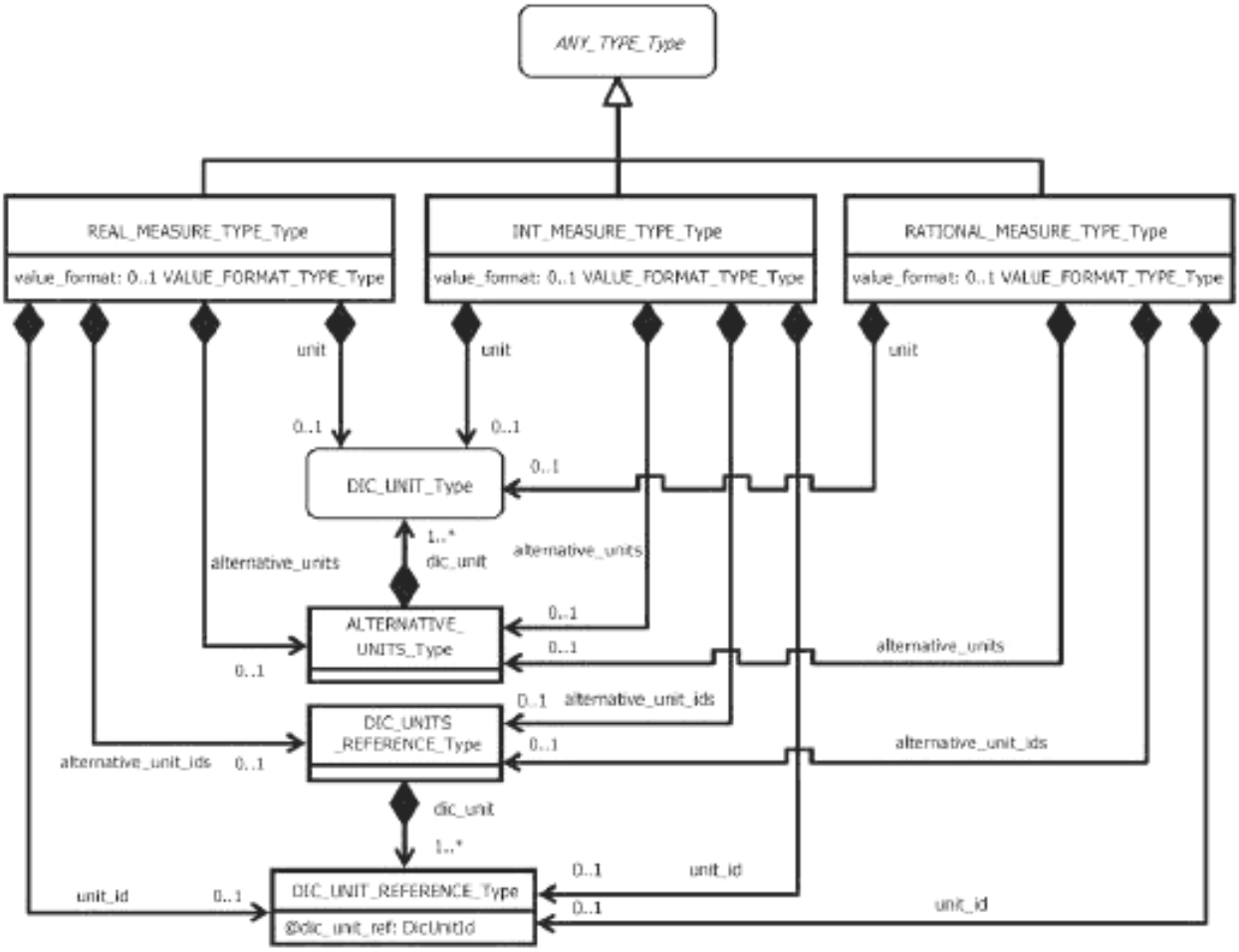


Рисунок 64 — Структура типов данных «численная мера»

Определения внутренних элементов:

Элемент **@dic_unit_ref** (тип данных **DIC_UNIT_REFERENCE_Type**): Определяет ссылку на словарную единицу.

Элементы **alternative_units** (типы данных **REAL_MEASURE_TYPE_Type, INT_MEASURE_TYPE_Type, RATIONAL_MEASURE_TYPE_Type**): Определяют список других единиц, которые могут использоваться для выражения свойства, чьей областью значений является мера.

Элементы **alternative_unit_ids** (типы данных **REAL_MEASURE_TYPE_Type, INT_MEASURE_TYPE_Type, RATIONAL_MEASURE_TYPE_Type**): Определяют список идентификаторов других единиц, которые могут использоваться для выражения свойства, чьей областью значений является мера.

Примечание 1 — Если значение свойства, чьей областью значений является мера, оценивается в единицах, определенных с помощью элемента **alternative_units** или **alternative_unit_ids**, то это значение меры может представляться как одиночное действительное значение, которое необходимо представлять в виде пары «значение/единица измерений».

Примечание 2 — Списочный порядок необходимо использовать для гарантии того, что элементы **alternative_units** и **alternative_unit_ids** (при условии, что они оба существуют) определяются одной и той же единицей измерения в одном и том же порядке.

Элемент **dic_unit** (тип данных **ALTERNATIVE_UNITS_Type**): Определяет описание альтернативной единицы меры.

Элемент **dic_unit** (тип данных **DIC_UNITS_REFERENCE_Type**): Определяет ссылку на альтернативную единицу меры.

Элементы **unit** (типы данных **REAL_MEASURE_TYPE_Type, INT_MEASURE_TYPE_Type, RATIONAL_MEASURE_TYPE_Type**): Определяют установленную по умолчанию ссылочную единицу, связанную с описываемой мерой.

Примечание 3 — Если значение свойства, чьей областью значений является мера, обменивается как единичное число, то это будет означать, что это значение выражается этой же определенной единицей меры.

Элементы **unit_id** (типы данных **REAL_MEASURE_TYPE_Type, INT_MEASURE_TYPE_Type, RATIONAL_MEASURE_TYPE_Type**): Определяют идентификатор установленной по умолчанию ссылочной единицы, связанной с описываемой мерой.

Примечание 4 — Если предоставляются оба элемента **unit** и **unit_id,** то элемент **unit** будет иметь преимущественную силу.

Примечание 5 — Если значение свойства, чьей областью значений является мера, обменивается как единичное число, то это будет означать, что это значение выражается в элементе **unit** или **unit_id** меры.

Элементы **value_format** (типы данных **REAL_MEASURE_TYPE_Type, INT_MEASURE_TYPE_Type, RATIONAL_MEASURE_TYPE_Type**): Определяет описание типа и длины рекомендуемого представления для отображения значения свойства. Данный атрибут (при его наличии) дает указание системе относительно способа отображения этого значения.

Примечание 6 — Элемент **value_format** не должен использоваться для ограничения определения типа численной меры.

Примечание 7 — Если элемент **value_format** несовместим со связанным с ним определением численной меры, то элемент **value_format** будет игнорироваться.

Примечание 8 — Если любое ограничительное условие на характеристику строки (см. 8.5.3.3.2) применимо к значению численной меры, то оно будет иметь преимущественную силу по сравнению с элементом **value_format**.

Определения внутренних типов:

Тип **ALTERNATIVE_UNITS_Type:** Является множеством допустимых альтернативных единиц меры.

Тип **DIC_UNIT_Type:** Является определением словарной единицы, см. 8.4.

Тип **DIC_UNITS_REFERENCE_Type:** Является определением множества ссылок на некоторые идентификаторы единиц меры.

Тип **DIC_UNIT_REFERENCE_Type:** Является определением ссылки на идентификатор единицы меры.

Примечание 9 — Идентификатор единицы меры создается в соответствии с правилами, определенными в ИСО/ТС 29002-5.

Тип **VALUE_FORMAT_TYPE_Type:** Является идентификатором значений, допустимых для использования в данном формате.

Примечание 10 — Значения для типа данных **VALUE_FORMAT_TYPE_Type** определены в соответствии с приложением H.

Перечень ограничительных условий:

Тип **INT_MEASURE_TYPE_Type:** Определяет значения свойств или типов пользователя, которые принадлежат типу целой меры.

Примечание 11 — Лексическое представление значения, чьим типом данных является тип **INT_MEASURE_TYPE_Type,** приведено в D.1.17 приложения D.

Тип **RATIONAL_MEASURE_TYPE_Type:** Приводит значения свойств или типов пользователя, которые принадлежат типу рациональной меры.

Пример — Диаметр винта: 4 1/8 дюймов.

Примечание 12 — Лексическое представление значения, чьим типом данных является тип **RATIONAL_MEASURE_TYPE_Type,** приведено в D.1.18 приложения D.

Тип **REAL_MEASURE_TYPE_Type:** Приводит значения свойств или типов пользователя, которые принадлежат типу действительной меры.

Примечание 13 — Лексическое представление значения, чьим типом данных является тип **REAL_MEASURE_TYPE_Type,** приведено в D.1.16 приложения D.

Определения внешних типов:

Тип **ANY_TYPE_Type:** См. 8.3.

Тип **DIC_UNIT_Type:** См. 8.4.

Перечень ограничительных условий:

Предоставляется либо элемент **unit,** либо элемент **unit_id,** либо оба элемента.

Если предоставляются группы элементов **alternative_units** и **alternative_unit_ids,** то они должны иметь одну и ту же длину.

Каждый элемент **dic_unit,** описываемый в группе элементов **alternative_units,** должен иметь элемент **string_representation**.

Значение для типа данных **VALUE_FORMAT_TYPE_Type** не должно превышать 80 символов.

8.3.8 Перечень типов целочисленных кодов

Целое число, которое должно принимать значение из множества перечислимых кодов, представляется с помощью комплексного XML-типа данных **NON_QUANTITATIVE_INT_TYPE_Type**. Каждый из этих кодов связывается с определенным содержанием (см. рисунок 65).

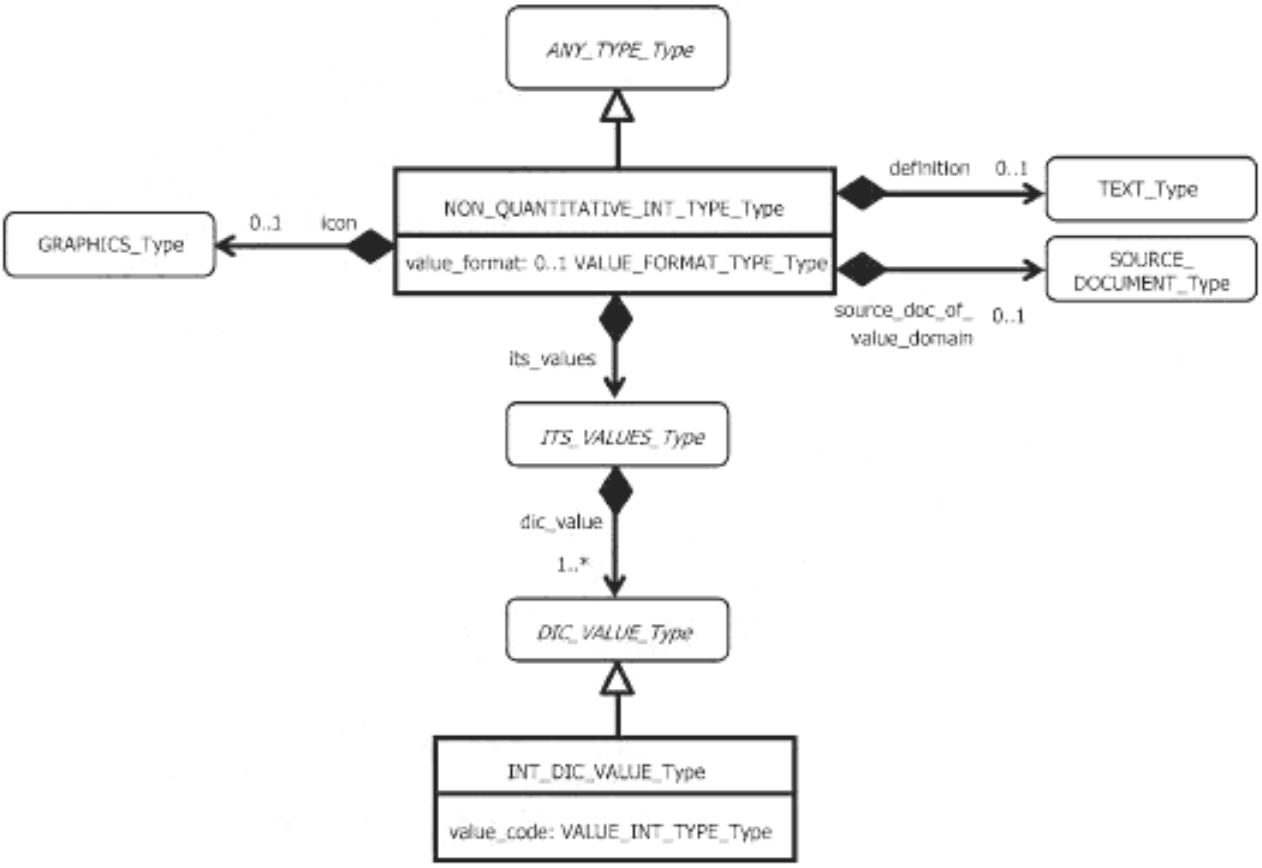


Рисунок 65 — Структура перечня типов целочисленных кодов

Перечисленные элементы определены в хранилище (контейнере) XML-элементов, называемых **its_values,** модель содержания которого определена с помощью комплексного XML-типа данных **ITS_VALUES_Type**. Каждый из этих элементов представляется посредством XML-элемента **dic_value,** собственная модель содержания которого определена типом данных **DIC_VALUE_Type,** или, более точно, в случае описания перечня целочисленных кодов с помощью специального подтипа комплексного XML-типа данных **INT_DIC_VALUE_Type**.

Определения внутренних элементов:

Элемент **definition:** Определяет текст, описывающий этот перечень (возможно, переведенный).

Элемент **icon:** Определяет пиктограммы, представляющие описание, которое связано с перечнем.

Элемент **its_values:** Определяет хранилище перечислимых элементов.

Элемент **its_values/dic_value/value_code:** Определяет целое значение перечислимого элемента.

Примечание 1 — Каждый XML-элемент **dic_value** модели содержания (определенный комплексным XML-типом данных **ITS_VALUES_Type**) представляется как комплексный XML-тип данных **INT_DIC_VALUE_Type**.

Элемент **source_doc_of_value_domain:** Определяет возможный документ-источник, из которого заимствовано определение перечня.

Элемент **value_format:** Определяет тип и длину рекомендуемого представления для отображения данного свойства. Данный атрибут (при его наличии) дает указание системе относительно способа отображения этого значения.

Примечание 2 — Элемент **value_format** не должен использоваться для ограничения определения типа численной меры.

Примечание 3 — Если элемент **value_format** несовместим со связанным с ним определением численной меры, то элемент **value_format** будет игнорироваться.

Примечание 4 — Если любое ограничительное условие на характеристику строки (см. 8.5.3.3.2) применимо к перечню типов целочисленных кодов, то оно будет иметь преимущественную силу по сравнению с элементом **value_format**.

Определения внутренних типов:

Тип **GRAPHICS_Type:** Является абстрактным комплексным XML-типом данных, представляющим внешний графический ресурс.

Тип **VALUE_FORMAT_TYPE_Type:** Является идентификатором значений, допускаемых для обозначения формата данных.

Примечание 5 — Значения для типа данных **VALUE_FORMAT_TYPE_Type** определены в соответствии с приложением H.

Перечень ограничительных условий:

Тип **NON_QUANTITATIVE_INT_TYPE_Type:** Предоставляет значения свойств или типов пользователя, которые являются перечнем типов целочисленных кодов.

Примечание 6 — Лексическое представление значения, чьим типом данных является тип **NON_QUANTITATIVE_INT_TYPE_Type**, приведено в D.1.19 приложения D.

Определения внешних типов:

Тип **ANY_TYPE_Type:** См. 8.3.

Тип **DIC_VALUE_Type:** См. 8.3.3.

Тип **ITS_VALUES_Type:** См. 8.3.3.

Тип **SOURCE_DOCUMENT_Type:** См. 8.2.2.1.

Тип **TEXT_Type:** См. раздел 8.1.2.

Перечень ограничительных условий:

Значение для типа данных **VALUE_FORMAT_TYPE_Type** не должно превышать 80 символов.

8.3.9 Типы групп элементов

Группы элементов предоставляются для определения типов данных, которые могут выражаться в виде списков, множеств, пакетов, массивов или ограниченных подмножеств любого вида значений. Различают пять видов групп: пакетов, множеств, списков, массивов и множеств с ограничительными условиями на их подмножества.

8.3.9.1 Группа элементов «пакет»

Пакет (bag) — это неупорядоченная группа (совокупность) значений, которая может содержать копии. Тип группы «пакет» представляется как комплексный XML-тип **BAG_TYPE_Type** (см. рисунок 66).

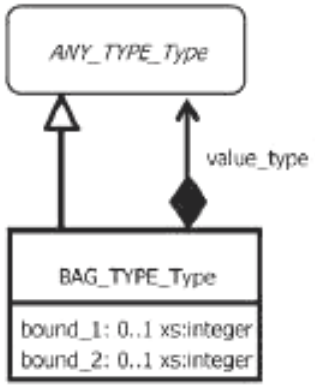


Рисунок 66 — Структура группы элементов «пакет»

Определения внутренних элементов:

Элемент **value_type:** Определяет тип значения (простого или сложного), который используется для каждого элемента пакета.

Элемент **bound_1:** Определяет возможное минимальное количество элементов в пакете.

Элемент **bound_2:** Определяет возможное максимальное количество элементов в пакете.

Определение внутреннего подтипа:

Подтип **BAG_TYPE_Type:** Опередяет значения свойств или типов пользователя, которые принадлежат типу неупорядоченных значений в группах, с возможными копиями.

Примечание — Лексическое представление значения, чьим типом данных является «пакет» **BAG_TYPE_Type,** приведено в D.1.20 приложения D.

Определение внешнего типа:

Тип **ANY_TYPE_Type:** См. 8.3.

Перечень ограничительных условий:

Если элемент **bound_1** определен, то он должен быть большим или равным нулю; в противном случае установленное по умолчанию значение будет равно нулю.

Если элемент **bound_2** определен, то он должен быть больше нуля; в противном случае установленное по умолчанию значение будет неизвестным (неограниченным).

Если оба элемента **bound_1** и **bound_2** определены, то элемент **bound_2** должен быть больше элемента **bound_1**.

8.3.9.2 Группа элементов «множество»

Множество (set) — это неупорядоченная группа (совокупность) значений, которая не может содержать копий дублированных значений, представляющихся как комплексный XML-тип данных **SET_TYPE_Type** (см. рисунок 67).

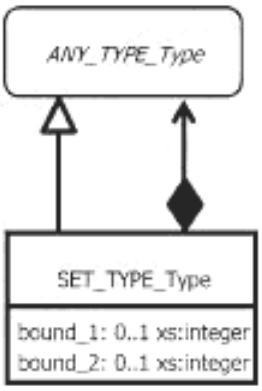


Рисунок 67 — Структура группы элементов «множество»

Определения внутренних элементов:

Элемент **value_type:** Определяет тип значения (простого или сложного), который используется для каждого элемента пакета.

Элемент **bound_1:** Определяет возможное минимальное количество элементов в множестве.

Элемент **bound_2:** Определяет возможное максимальное количество элементов в множестве.

Определение внутреннего подтипа:

Тип **SET_TYPE_Type:** Определяет значения свойств или типов пользователя, которые принадлежат типу неупорядоченных значений в группах, без копий.

Примечание — Лексическое представление значения, чьим типом данных является «множество» **SET_TYPE_Type,** приведено в D.1.21 приложения D.

Определение внешнего типа:

Тип **ANY_TYPE_Type**: См. 8.3.

Перечень ограничительных условий:

Если элемент **bound_1** определен, то он должен быть больше или равным нулю; в противном случае установленное по умолчанию значение будет равно нулю.

Если элемент **bound_2** определен, то он должен быть больше нуля; в противном случае установленное по умолчанию значение будет неизвестным (не ограниченным).

Если оба элемента **bound_1** и **bound_2** определены, то элемент **bound_2** должен быть больше элемента **bound_1**.

8.3.9.3 Группа элементов «список»

Список (list) — это упорядоченная группа (совокупность) значений, которые могут быть уникальными. Тип группы «список» представляется как комплексный XML-тип данных **LIST_TYPE_Type** (см. рисунок 68).

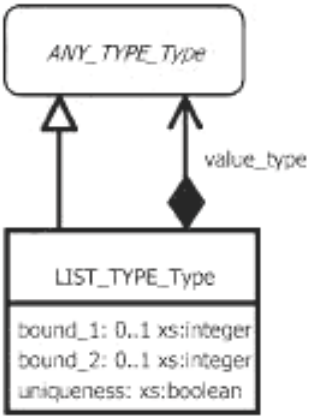


Рисунок 68 — Структура группы элементов «список»

Определения внутренних элементов:

Элемент **value_type:** Определяет тип значения (простого или сложного), который используется для каждого элемента списка.

Элемент **bound_1:** Определяет возможное минимальное количество элементов в списке.

Элемент **bound_2:** Определяет возможное максимальное количество элементов в списке.

Элемент **uniqueness:** Определяет метку элемента булевого типа, предназначенную для указания того, являются ли все элементы списка уникальными (истинное значение (true)) или допускаются дублированные элементы (ложное значение (false)).

Перечень ограничительных условий:

Тип **LIST_TYPE_Type:** Приводит значения свойств или типов пользователя, которые принадлежат типу неупорядоченных значений в группах, возможно, уникальных.

Примечание — Лексическое представление значения, чьим типом данных является «список» **LIST_TYPE_Type**, приведено в D.1.22 приложения D.

Определение внешнего типа:

Тип **ANY_TYPE_Type:** См. 8.3.

Перечень ограничительных условий:

Если элемент **bound_1** определен, то он должен быть больше или равным нулю; в противном случае установленное по умолчанию значение будет равно нулю.

Если элемент **bound_2** определен, то он должен быть больше нуля; в противном случае установленное по умолчанию значение будет неизвестным (неограниченным).

Если оба элемента **bound_1** и **bound_2** определены, то элемент **bound_2** должен быть больше элемента **bound_1**.

8.3.9.4 Группа элементов «массив»

Массив (array) — это упорядоченная группа (совокупность) фиксированной длины из значений, которые могут быть уникальными или произвольными, но маркированными последовательными индексами из целых чисел. Тип «массив» представляется как комплексный XML-тип данных **ARRAY_TYPE_Type** (см. рисунок 69).

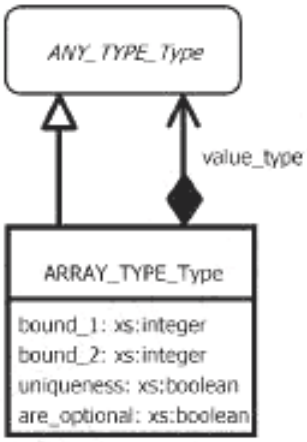


Рисунок 69 — Структура группы элементов «массив»

Определения внутренних элементов:

Элемент **value_type:** Определяет тип значения (простого или сложного), который используется для каждого элемента массива.

Элемент **bound_1:** Определяет целое число, указывающее минимальный индекс определенного типа массива.

Элемент **bound_2:** Определяет целое число, указывающее максимальный индекс определенного типа массива.

Элемент **uniqueness:** Определяет метку элемента булевого типа, предназначенную для указания того, должны ли присутствовать все элементы в массиве (ложное значение (false)) или некоторые элементы могут отсутствовать (истинное значение (true)).

Элемент **are_optional:** Определяет метку элемента булевого типа, предназначенную для указания того, должны ли присутствовать все элементы в массиве (ложное значение (false)) или некоторые элементы могут отсутствовать (истинное значение (true)).

Перечень ограничительных условий:

Тип **ARRAY_TYPE_Type:** Приводит значения свойств или типов пользователя, которые принадлежат типу упорядоченных и проиндексированных значений, возможно, уникальных и произвольных.

Примечание — Лексическое представление значения, чьим типом данных является «массив» **ARRAY_TYPE_Type,** приведено в D.1.23 приложения D.

Определение внешнего типа:

Тип **ANY_TYPE_Type:** См. 8.3.

Перечень ограничительных условий:

Элемент **bound_1** должен быть меньшим или равным элементу **bound_2**.

8.3.9.5 Группа элементов «множество с ограничительным условием на подмножество»

Множество с ограничительным условием на подмножество (set with subset constraint) — это неупорядоченная группа (совокупность), не содержащая копий (дублей), из которой может извлекаться подмножество от минимального до максимального количества элементов (объема). Эта группа представляется как комплексный XML-тип данных **SET_WITH_SUBSET_CONSTRAINT_TYPE_Type** (см. рисунок 70).

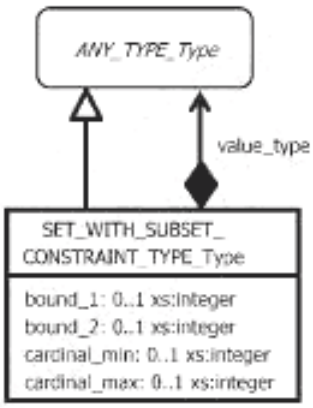


Рисунок 70 — Структура группы элементов «множество с ограничительным условием на подмножество»

Определения внутренних элементов:

Элемент **value_type:** Определяет тип значения (простого или сложного), который используется для каждого элемента множества с ограничительным условием на подмножество.

Элемент **bound_1:** Определяет целое число, указывающее минимальный индекс определенного множества с ограничительным условием на подмножество.

Элемент **bound_2:** Определяет целое число, указывающее максимальный индекс определенного множества с ограничительным условием на подмножество.

Элемент **cardinal_min:** Определяет минимальный объем подмножества, который может быть извлечен.

Элемент **cardinal_max:** Определяет максимальный объем подмножества, который может быть извлечен.

Перечень ограничительных условий:

Тип **SET_WITH_SUBSET_CONSTRAINT_TYPE_Type:** Определяет значения свойств или типов пользователя, которые принадлежат типу неупорядоченных значений в группах, не содержат дублей и из которых может извлекаться подмножество от минимального до максимального объема.

Примечание — Лексическое представление значения, чьим типом данных является «множество с ограничительным условием на подмножество» **SET_WITH_SUBSET_CONSTRAINT_Type**, приведено в D.1.24 приложения D.

Определение внешнего типа:

Тип **ANY_TYPE_Type:** См. 8.3.

Перечень ограничительных условий:

Если элемент **bound_1** определен, то он должен быть больше или равным нулю; в противном случае установленное по умолчанию значение будет равно нулю.

Если элемент **bound_2** определен, то он должен быть больше нуля; в противном случае установленное по умолчанию значение будет неизвестным (неограниченным).

Если оба элемента **bound_1** и **bound_2** определены, то элемент **bound_2** должен быть больше элемента **bound_1**.

Элемент **cardinal_min** должен быть меньше или равным элементу **cardinal_max**.

Если элементы **bound_1** и **cardinal_min** определены, то элемент **cardinal_min** не должен превышать элемент **bound_1**.

Если элементы **bound_2** и **cardinal_max** определены, то элемент **cardinal_max** не должен превышать элемент **bound_2**.

8.3.10 Тип «ссылочный класс»

Тип ссылочного класса (class reference) позволяет определять область значений свойства, которыми являются экземпляры класса.

Примечание 1 — Свойство, чьим типом является тип ссылочного класса, устанавливает соотношение между обоими классами. Таким соотношением может быть, например, композиционное соотношение.

Пример — Сборка bolted assembly может содержать элементы screw и nut. Предположим, что bolted assembly, screw и nut характеризуют семейство деталей; при этом компоненты bolted assembly могут представляться, во-первых, путем определения двух свойств в классе bolted assembly (соответственно — элементов its_screw и its_nut), и, во-вторых, путем закрепления за этими свойствами области значений, которые будут отвечать типу данных CLASS_REFERENCE_TYPE_Type, связывая классы screw и nut.

Тип ссылочного класса представляется как комплексный XML-тип данных **CLASS_REFERENCE_TYPE_Type** (см. рисунок 71).

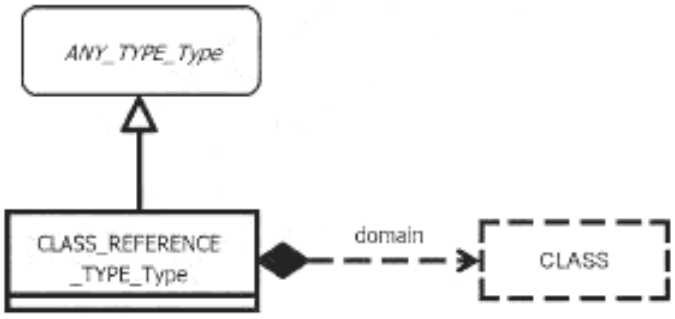


Рисунок 71 — Структура области значений экземпляра класса

Определение внутреннего элемента:

Элемент **domain:** Определяет ссылку на онтологическое понятие класса, которое определяет область значений для типа данных.

Перечень ограничительных условий:

Тип **CLASS_REFERENCE_TYPE_Type:** Приводит значения свойств или типов пользователя, которые принадлежат типу экземпляров класса.

Примечание 2 — Лексическое представление значения, чьим типом данных является тип **SET_TYPE_Type**, приведено в D.1.25 приложения D.

Определение внешнего типа:

Тип **ANY_TYPE_Type:** См. 8.3.

8.3.11 Тип данных «уровень»

Тип данных «уровень» (level) является комплексным и указывающим, что значение свойства состоит (из) от одной до четырех действительных или целочисленных мер, определяющих характеристику элемента в фиксированной последовательности значений: минимального (**min**), номинального (**nom**), типового (**type**) и максимального (**max**) значений, т. е.:

- **min:** наименьшая величина, выраженная количественно и установленная для определенного набора рабочих условий, при которых компонент, устройство или оборудование будут находиться в рабочем состоянии и отвечать установленным требованиям;
- **nom:** количественно выраженная величина, используемая для обозначения и идентификации компонента, устройства или оборудования;
- **type:** наиболее часто встречающаяся количественно выраженная величина, используемая для целей определения характеристик и устанавливаемая для определенного набора рабочих условий, при которых компонент, устройство или оборудование будут находиться в рабочем состоянии;
- **max:** наибольшее значение, выраженное количественно и установленное для определенного набора рабочих условий, при которых компонент, устройство или оборудование будут находиться в рабочем состоянии и отвечать установленным требованиям.

Примечание 1 — Номинальное значение обычно является округленной величиной.

Пример — Автомобильный аккумулятор с номинальным напряжением 12 В имеет 6 ячеек с типовым напряжением примерно 2,2 В, что дает типовое значение напряжения аккумулятора примерно 13,5 В. При зарядке максимальное напряжение аккумулятора может достигать примерно 14,5 В, однако он будет считаться полностью разряженным, если его напряжение упадет до минимального значения 12,5 В.

Примечание 2 — Рекомендуется, чтобы использование типа данных «уровень» ограничивалось теми свойствами в области значений, где сообщение о многих значениях единственной характеристики считается общепринятой и запрашиваемой методикой и часто используется на производстве электронных компонентов.

Этот тип представляется как комплексный XML-тип данных **LEVEL_TYPE_Type** (см. рисунок 72).

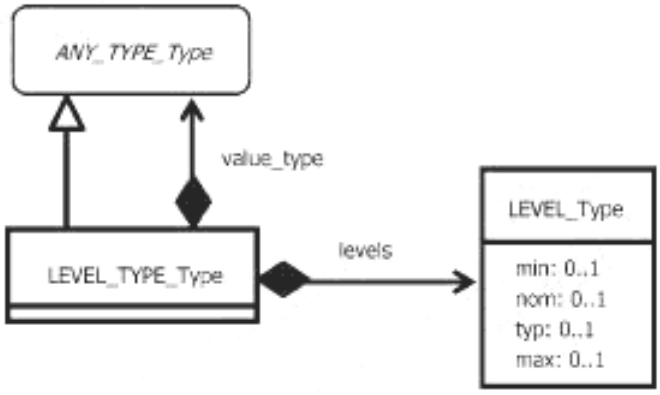


Рисунок 72 — Структура области значений уровней

Определение внутреннего элемента:

Элемент **levels:** Определяет список квалификаторов, который связан со свойством.

max: максимальный квалификатор, закрепленный за областью значений свойства.

min: минимальный квалификатор, закрепленный за областью значений свойства.

nom: номинальный квалификатор, закрепленный за областью значений свойства.

typ: типовой квалификатор, закрепленный за областью значений свойства.

Элемент **value_type:** Определяет тип значения, который связывается с описанием уровней.

Примечание 3 — Какая-либо модель содержания, связанная с определенными квалификаторами, отсутствует, поэтому никакой тип данных не закрепляется за XML-элементами, соответствующими каждому квалификатору.

Определение внутреннего типа:

Тип **LEVEL_Type:** Является описанием возможных уровней.

Примечание 4 — Каждый уровень представляется дополнительным и пустым XML-элементом.

Перечень ограничительных условий:

Тип **LEVEL_TYPE_Type:** Определяет значения свойств или типов пользователя, которые принадлежат типу квалифицированных числовых значений.

Примечание 5 — Лексическое представление значения, чьим типом данных является «уровень» **LEVEL_TYPE_Type,** приведено в D.1.26 приложения D.

Определение внешнего типа:

Тип **ANY_TYPE_Type:** См. 8.3.

Перечень ограничительных условий:

Основополагающим типом данных, определенным с помощью XML-элемента **value_type,** является тип числовой меры, определенный в 8.3.7.

Определяется как минимум один квалификатор.

8.3.12 Именованный тип данных

Именованный тип данных (named) позволяет представлять область значений в виде ссылки на идентифицированное онтологическое понятие типа данных (см. 6.7.6). Этот тип представляется с помощью комплексного XML-типа данных **NAMED_TYPE_Type** (см. рисунок 73).

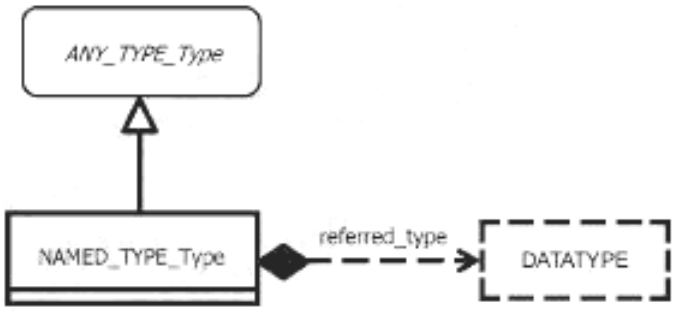


Рисунок 73 — Структура именного типа данных

Определение внутреннего элемента:

Элемент **referred_type:** Определяет ссылку на онтологическое понятие типа данных, определяющее область значений.

Перечень ограничительных условий:

Тип **NAMED_TYPE_Type:** Определяет значения свойств или типов пользователя, которые принадлежат любому виду OntoML-типа данных, определенному с помощью ссылочного типа данных.

Примечание — Лексическое представление значения, чьим типом данных является **NAMED_TYPE_Type,** является лексическим представлением исходного типа данных.

Определение внешнего типа:

Тип **ANY_TYPE_Type:** См. 8.3.

8.3.13 Типы данных повышенного уровня

В системе OntoML-типов определены типы данных, которые могут использоваться для определения характеристик областей значений комплексного свойства при описании онтологии продукции.

Эти типы данных определены в онтологии, которая является частью настоящего стандарта.

Примечание 1 — Онтология типов данных повышенного уровня определена в приложении E.

Таким образом, каждый тип данных повышенного уровня (advanced-level data) определяется путем ссылки на онтологический класс, который определяется его структурой значений. Значения свойств могут после этого представляться и участвовать в обмене в соответствии с ИСО/ТС 29002-10.

Примечание 2 — ИСО/ТС 29002 разрабатывается объединенными усилиями нескольких технических комитетов по стандартизации с целью облегчения установления функциональной совместимости между различными стандартами, в которых требуются определения характеристик продукции.

Расширенные типы данных, поддерживаемые OntoML-языком, иллюстрируются на рисунке 74.

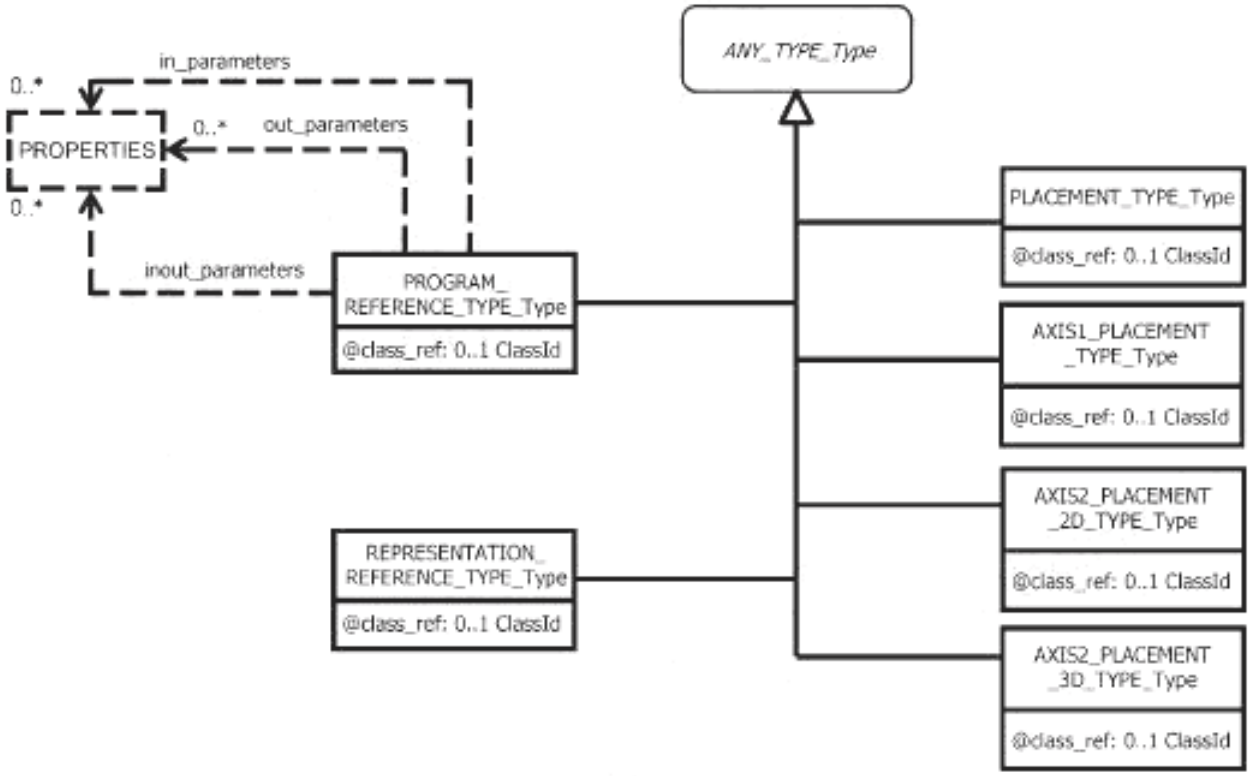


Рисунок 74 — Структура типов данных повышенного уровня

Определения внутренних элементов:

Элемент **@class_ref (AXIS1_PLACEMENT_TYPE_Type):** Определяет возможную ссылку на класс онтологии, который определяет структуру типа данных **AXIS1_PLACEMENT_TYPE_Type**.

Примечание 3 — Если этот элемент предоставляется, то XML-атрибут **@class_ref** должен устанавливаться равным "0112-1---13584_32_1#01-AXIS1_PLACEMENT#1" IRDI.

Элемент **@class_ref (AXIS2_PLACEMENT_2D_TYPE_Type):** Определяет возможную ссылку на класс онтологии, который определяет структуру типа данных **AXIS2_PLACEMENT_2D_TYPE_Type**.

Примечание 4 — Если этот элемент предоставляется, то XML-атрибут **@class_ref** должен устанавливаться равным "0112-1---13584_32_1#01-AXIS2_PLACEMENT_2D#1" IRDI.

Элемент **@class_ref (AXIS2_PLACEMENT_3D_TYPE_Type):** Определяет возможную ссылку на класс онтологии, который определяет структуру типа данных **AXIS2_PLACEMENT_3D_TYPE_Type**.

Примечание 5 — Если этот элемент предоставляется, то XML-атрибут **@class_ref** должен устанавливаться равным "0112-1---13584_32_1#01-AXIS2 PLACEMENT 3D#1" IRDI.

Элемент **@class_ref (PLACEMENT_TYPE_Type):** Определяет возможную ссылку на класс онтологии, который определяет структуру типа данных **PLACEMENT_TYPE_Type**.

Примечание 6 — Если этот элемент предоставляется, то XML-атрибут **@class_ref** должен устанавливаться равным "0112-1---13584_32_1#01-PLACEMENT#1" IRDI.

Элемент **@class_ref (PROGRAM_REFERENCE_TYPE_Type):** Определяет возможную ссылку на класс онтологии, который определяет структуру типа данных **PROGRAM_REFERENCE_TYPE_type**.

Примечание 7 — Если этот элемент предоставляется, то XML-атрибут **@class_ref** должен устанавливаться равным "0112-1---13584_32_1#01-PROGRAM_REFERENCE#1" IRDI.

Элемент **@class_ref (REPRESENTATION_REFERENCE_TYPE_Type):** Определяет возможную ссылку на класс онтологии, который определяет структуру типа данных **REPRESENTATION_REFERENCE_TYPE_Type**.

Примечание 8 — Если этот элемент предоставляется, то XML-атрибут **@class_ref** должен устанавливаться равным "0112-1---13584_32_1#01-REPRESENTATION_REFERENCE#1" IRDI.

Элемент **in_parameters:** Определяет список ссылок на свойство, который определяет тип входных параметров ссылочной программы.

Элемент **inout_parameters:** Определяет список ссылок на свойство, который определяет тип входных/выходных параметров ссылочной программы.

Элемент **out_parameters:** Определяет список ссылок на свойство, который определяет тип выходных параметров ссылочной программы.

Перечень ограничительных условий:

Тип **AXIS1_PLACEMENT_TYPE_Type:** Позволяет определять область значений свойства, чье значение является экземпляром онтологического класса ***axis 1 placement,*** который определяет направление и местоположение одиночной оси в трехмерной области.

Примечание 9 — Лексическое представление значения, чьим типом данных является **LEVEL_TYPE_Type,** приведено в D.3.1.2 приложения D.

Примечание 10 — Онтологический класс ***axis 1 placement*** с идентификатором «0112-1---13584_32_1#01-AXIS1_PLACEMENT#1» IRDI определяется в онтологии компонентов для значений типов данных повышенного уровня, указанной в приложении E.

Примечание 11 — Компонент онтологического класса ***axis 1 placement*** соответствует компоненту типа объектных EXPRESS-данных ***axis1_placement,*** определенному в ИСО 10303-42.

Тип **AXIS2_PLACEMENT_2D_TYPE_Type:** Позволяет определять область значений свойства, чье значение является экземпляром онтологического класса ***axis 2 placement 2D,*** который определяет ориентацию и местоположение двух взаимно перпендикулярных осей в двухмерной области.

Примечание 12 — Лексическое представление значения, чьим типом данных является **AXIS2_PLACEMENT_2D_TYPE_Type,** приведено в D.3.1.3 приложения D.

Примечание 13 — Онтологический класс ***axis_2 placement 2D*** с идентификатором «0112-1---13584_32_1#01-AXIS2_PLACEMENT_2D#1» IRDI определяется в онтологии компонентов для значений типов данных повышенного уровня, рассмотренной в приложении E.

Примечание 14 — Компонент онтологического класса ***axis_2 placement 2D*** соответствует компоненту типа объектных EXPRESS-данных ***axis2_placement_2d,*** определенному в ИСО 10303-42.

Тип **AXIS2_PLACEMENT_3D_TYPE_Type:** Позволяет определять область значений свойства, чье значение является экземпляром онтологического класса ***axis 2 placement 3D,*** который определяет ориентацию и местоположение двух взаимно перпендикулярных осей в трехмерной области.

Примечание 15 — Лексическое представление значения, чьим типом данных является **AXIS2_PLACEMENT_3D_TYPE_Type,** приведено в D.3.1.4 приложения D.

Примечание 16 — Онтологический класс ***axis_2 placement 3D*** с идентификатором "0112-1---13584_32_1#01-AXIS2_PLACEMENT_3D#1" IRDI определяется в онтологии компонентов для значений типов данных повышенного уровня, рассмотренной в приложении E.

Примечание 17 — Компонент онтологического класса ***axis_2 placement 3D*** соответствует компоненту типа объектных EXPRESS-данных ***axis2_placement_3d,*** определенному в ИСО 10303-42.

Тип **PLACEMENT_TYPE_Type:** Позволяет определять область значений свойства, чье значение является экземпляром онтологического класса ***placement,*** который определяет положение по отношению к координатной системе его геометрического вида.

Примечание 18 — Лексическое представление значения, чьим типом данных является **PLACEMENT_TYPE_Type,** приведено в D.3.1.1 приложения D.

Примечание 19 — Онтологический класс ***placement*** с идентификатором "0112-1---13584_32_1#01-PLACEMENT#1" IRDI определяется в онтологии компонентов для значений типов данных повышенного уровня, рассмотренной в приложении E.

Примечание 20 — Компонент онтологического класса ***placement*** соответствует компоненту типа объектных EXPRESS-данных ***placement,*** определенному в ИСО 10303-42.

Тип **PROGRAM_REFERENCE_TYPE_Type:** Позволяет определять область значений свойства, чье значение является экземпляром онтологического класса ***program reference,*** содержащего алгоритм, который должен активироваться и предоставляться вместе со значениями параметров для создания каждого представления элемента.

Примечание 21 — Лексическое представление значения, чьим типом данных является **PROGRAM_REFERENCE_TYPE_Type,** приведено в D.3.2.2 приложения D.

Примечание 22 — Онтологический класс ***program reference*** с идентификатором "0112-1---13584_32_1#01-PROGRAM_REFERENCE#1" IRDI определяется в онтологии компонентов для значений типов данных повышенного уровня, рассмотренной в приложении E.

Примечание 23 — Компонент онтологического класса ***program reference*** соответствует компоненту типа объектных EXPRESS-данных ***program_reference,*** определенному в ИСО 13584-25.

Тип **REPRESENTATION_REFERENCE_TYPE_Type:** Позволяет определять область значений свойства, чье значение является экземпляром онтологического класса ***representation reference***.

Примечание 24 — Лексическое представление значения, чьим типом данных является **REPRESENTATION_REFERENCE_TYPE_Type,** приведено в D.3.2.1 приложения D.

Примечание 25 — Онтологический класс **representation reference** с идентификатором "0112-1---13584_32_1#01-REPRESENTATION_REFERENCE#1" IRDI, определяется в онтологии компонентов для значений типов данных повышенного уровня, указанной в приложении E.

Примечание 26 — Компонент онтологического класса ***representation reference*** соответствует компоненту типа объектных EXPRESS-данных ***representation reference,*** определенному в ИСО 13584-25.

Определение вешнего типа:

Тип **ANY_TYPE_Type:** См. 8.3.

8.4 Единицы измерений

Свойства, для которых тип данных представляет количественные показатели, связаны с единицами измерений.

Примечание 1 — Единицы измерений в OntoML-языке определены в соответствии с ИСО 10303-41 для информационной модели представления единиц измерений.

В OntoML-языке единица измерений свойства представляется с помощью комплексного XML-типа данных **DIC_UNIT_Type** (см. рисунок 75).

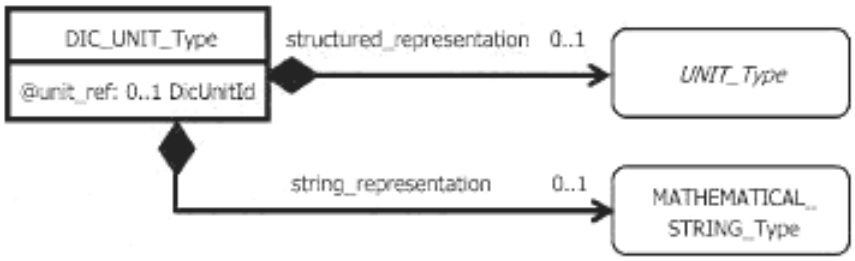


Рисунок 75 — Общая структура единиц измерения свойства

Определение внутреннего элемента:

Элемент **@unit_ref:** Определяет возможную ссылку на идентификатор единицы измерений.

Элемент **string_representation:** Определяет строковое представление единицы измерений свойства.

Элемент **structured_representation:** Определяет явное описание единицы измерений свойства.

Примечание 2 — Если предоставляется ссылка на единицу измерений и ее структурированное представление, то в случае возникновения противоречий представление будет иметь приоритет.

Определения внешних типов:

Тип **MATHEMATICAL_STRING_Type:** Является представлением математической строки, см. 8.8.2.

Тип **UNIT_Type:** Является описанием единицы измерений, см. 8.4.1.

Перечень ограничительных условий:

Должна предоставляться либо ссылка на единицу измерений (элемент **unit_ref**), либо ее четкое представление (элемент **structured_representation**), либо и то и другое.

8.4.1 Структура единицы измерений

Единица измерений определяется с помощью абстрактного комплексного XML-типа данных **UNIT_Type** (см. рисунок 76).

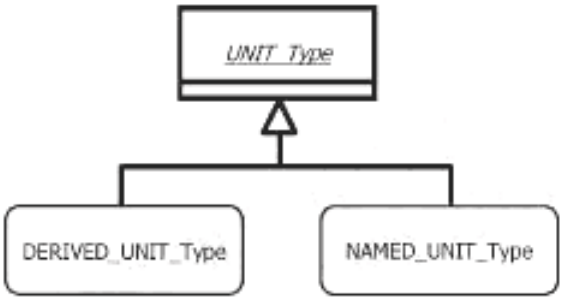


Рисунок 76 — Основные структуры единиц измерений

Определения внешних типов:

Тип **DERIVED_UNIT_Type:** Является типом производной единицы измерения, см. 8.4.3.

Пример 1 — Единица давления Н/мм^2 является производной единицей измерения.

Тип **NAMED_UNIT_Type:** Является типом именованной единицы измерения, см. 8.4.2.

Пример 2 — Единицы измерений мм или Па являются видами именованных единиц измерения.

8.4.2 Именованные единицы измерений

Именованные единицы измерений — это единицы, связанные со словами или группами слов, с помощью которых они могут быть идентифицированы. Эти единицы представляются с помощью комплексного XML-типа данных **NAMED_UNIT_Type** (см. рисунок 77).

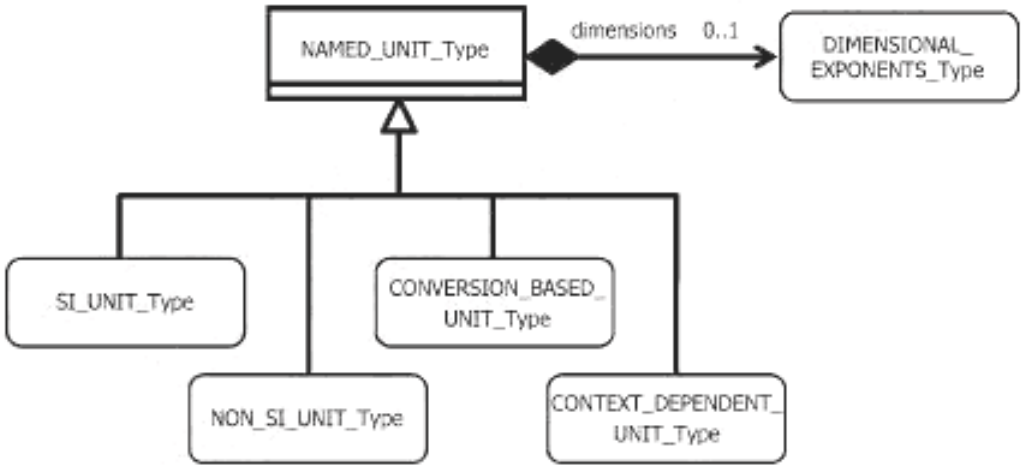


Рисунок 77 — Общая структура именованных единиц измерений

Определение внутреннего элемента:

Элемент **dimensions:** Определяет возможные экземпляры основных свойств, с помощью которых определяется именованная единица измерения.

Определение внутреннего типа:

Тип **DIMENSIONAL_EXPONENTS_Type:** Является уравнением (соотношением) размерностей, см. 8.4.2.1.

Определения внешних типов:

Тип **CONTEXT_DEPENDENT_UNIT_Type:** Является контекстно-зависимой единицей измерения, см. 8.4.2.5.

Тип **CONVERSION_BASED_UNIT_Type:** Является преобразованной единицей измерения, см. 8.4.2.4.

Тип **NON_SI_UNIT_Type:** Является международно-нестандартизированной единицей измерения, см. 8.4.2.3.

Тип **SI_UNIT_Type:** Является международно-стандартизированной единицей измерения, см. 8.4.2.2.

8.4.2.1 Показатель размерности

Именованные единицы измерений могут быть связаны с их уравнением размерностей и представляться с помощью комплексного XML-типа данных **DIMENSIONAL_EXPONENTS_Type** (см. рисунок 78).

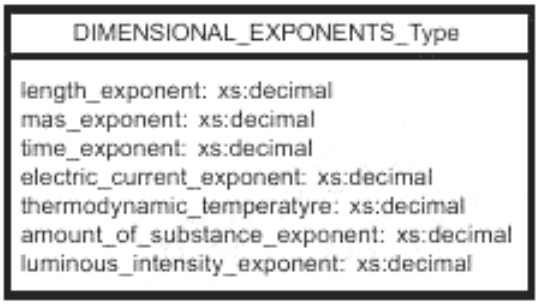


Рисунок 78 — Структура показателя размерности

Определения внутренних элементов:

Элемент **amount_of_substance_exponent:** Определяет показатель степени для значения основной вещественной величины.

Элемент **electric_current_exponent:** Определяет показатель степени для значения основной единицы электрического тока.

Элемент **length_exponent:** Определяет показатель степени для значения основной единицы длины.

Элемент **luminous_intensity_exponent:** Определяет показатель степени для значения основной единицы яркости.

Элемент **mass_exponent:** Определяет показатель степени для значения основной единицы массы.

Элемент **thermodynamic_temperature:** Определяет показатель степени для значения основной единицы термодинамической температуры.

Элемент **time_exponent:** Определяет показатель степени для значения основной единицы времени.

8.4.2.2 Международно-стандартизированная единица измерений

Международно-стандартизированная единица измерений — это зафиксированная величина, используемая в качестве стандартной при выражении тех элементов, которые измеряются в соответствии с ИСО 80000-1.

Пример 1 — Миллиметр является международно-стандартизированной единицей измерения.

Международно-стандартизированная единица измерений определяется посредством дополнительного префикса и связывается с соответствующей единицей в системе единиц СИ. Эта единица представляется с помощью комплексного XML-типа данных **SI_UNIT_Type** (см. рисунок 79).

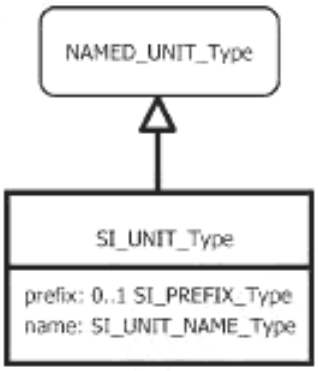


Рисунок 79 — Структура международно-стандартизированной единицы измерений

Определения внутренних элементов:

Элемент **name:** Определяет слово или группу слов, с помощью которых дается ссылка на международно-стандартизированную единицу измерений.

Пример 2 — Миллиметр является единицей системы СИ: В соответствии с описанием комплексного XML-типа данных SI_UNIT_Type значение "MILLI" будет присваиваться дополнительному префиксу XML-элемента, а значение "METRE" — обязательному имени XML-элемента.

Элемент **prefix:** Определяет международно-стандартизированный префикс единицы измерений.

Определения внутренних типов:

Тип **SI_PREFIX_Type:** Является именем префикса, который может связываться с международно-стандартизированной единицей измерения и представляется с помощью перечня допускаемых международно-стандартизированных префиксов единиц измерений.

Примечание 1 — Допустимые международно-стандартизированные префиксы единиц измерений определены в ИСО 80000-1.

Тип **SI_UNIT_NAME_Type:** Является именем международно-стандартизированной единицы измерений и представляется с помощью перечня допустимых международно-стандартизированных имен единиц измерений.

Примечание 2 — Допустимые международно-стандартизированные имена единиц измерений определены в ИСО 80000-1.

Определение внешнего типа:

Тип **NAMED_UNIT_Type:** См. 8.4.2.

Перечень ограничительных условий:

Для международно-стандартизированной единицы измерений наследуемый атрибут **dimensions** не используется.

8.4.2.3 Международно-нестандартизированная единица измерений

Международно-нестандартизированная единица измерений позволяет представлять единицы измерений, которые не принадлежат системе единиц СИ, не являются преобразованными единицами или единицами длины. Эти единицы представляются с помощью комплексного XML-типа данных **NON_SI_UNIT_Type** (см. рисунок 80).

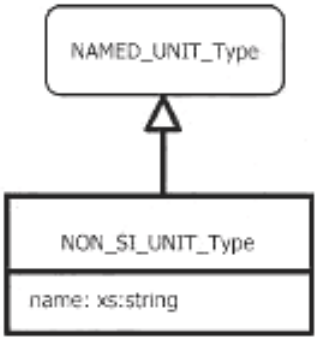


Рисунок 80 — Структура международно-нестандартизированной единицы измерений

Определение внутреннего элемента:

Элемент **name:** Определяет метку, используемую для наименования рассматриваемой единицы измерений.

Определение внешнего типа:

Тип **NAMED_UNIT_Type:** См. 8.4.2.

8.4.2.4 Преобразованная единица измерений

Преобразованная единица измерений — это единица, определенная на основе другой единицы измерений.

Пример 1 — Дюйм является преобразованной единицей измерения.

Эта единица представляется с помощью комплексного XML-типа данных **CONVERSION_BASED_UNIT_Type** (см. рисунок 81).

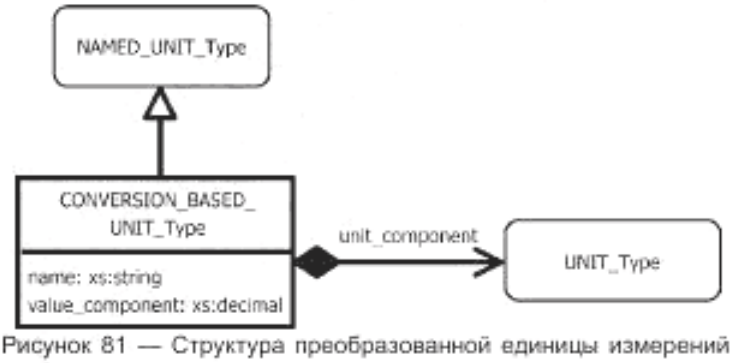


Рисунок 81 — Структура преобразованной единицы измерений

Определения внутренних элементов:

Элемент **name:** Определяет слово или группу слов, с помощью которых преобразованная единица измерений имеет ссылку на основную единицу.

Элемент **unit_component:** Определяет единицу измерений, с помощью которой выражается физическая величина.

Элемент **value_component:** Определяет значение физической величины, которая соответствует одной преобразованной единице измерений, если она выражается с помощью элемента **unit_component** единицы измерений.

Пример 2 — Дюйм является преобразованной единицей измерения, заимствованной из британской системы мер и весов, имеющей имя XML-элемента, эквивалентное "inch", которое может связываться с единицей системы СИ — миллиметром — посредством значения меры (XML-элемент value_component), равного 25,4 мм (XML-элемент unit_component).

Определения внешних типов:

Тип **NAMED_UNIT_Type:** См. 8.4.2.

Тип **UNIT_Type:** Является общей единицей, см. 8.4.

8.4.2.5 Контекстно-зависимая единица измерений

Контекстно-зависимая единица измерений — это единица, которая не связана с единицами системы СИ и представляется с помощью комплексного XML-типа данных **CONTEXT_DEPENDENT_UNIT_Type** (см. рисунок 82).

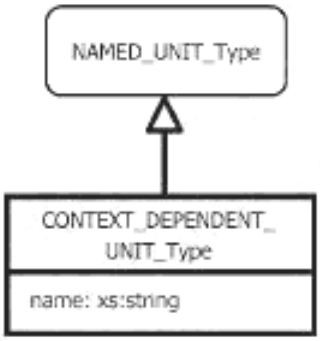


Рисунок 82 — Структура контекстно-зависимой единицы измерений

Определение внутреннего элемента:

Элемент **name:** Определяет слово или группу слов, с помощью которых дается ссылка на контекстно-зависимую единицу измерений.

Пример — Номер детали в сборочном узле является физической величиной, измеряемой в единицах, называемых «деталями», однако которые не могут быть связаны с единицами системы СИ. Значение «деталь» затем может присваиваться контекстно-зависимому имени единицы измерений.

Определение внешнего типа:

Тип **NAMED_UNIT_Type:** См. 8.4.2.

8.4.3 Производная единица измерений

Производная единица выражает единицы измерений.

Пример 1 — Единица Н/мм^2 является производной единицей измерения.

Эта единица представляется с помощью комплексного XML-типа данных **DERIVED_UNIT_Type** (см. рисунок 83).

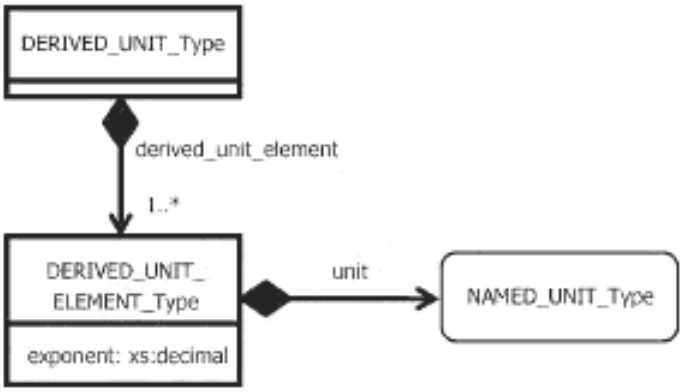


Рисунок 83 — Структура производной единицы измерений

Определения внутренних элементов:

Элемент **derived_unit_element:** Определяет единичную величину, которая составляет производную единицу измерений.

Элемент **derived_unit_element/exponent:** Определяет показатель степени, применимый к XML-элементу единицы измерений.

Элемент **derived_unit_element/unit:** Определяет фиксированную величину, которая используется в качестве математического множителя.

Пример 2 — Единица Newton per square millimeter является производной единицей, которая затем будет представляться с помощью двух элементов derived_unit_elements: первого — для представления свойства Newton как единицы системы СИ, и второго — для представления свойства millimeter как единицы системы СИ, которой будет присваиваться степень — 2.

Определение внешнего типа:

Тип **NAMED_UNIT_Type:** См. 8.4.2.

8.5 Ограничительные условия

В OntoML-языке ограничительные условия позволяют дополнительно ограничивать сопряженную область любого свойства, определенного в онтологии. Эти условия также используются для ограничения сопряженных областей, когда они определены в заданном классе и когда область для свойства является подклассом данного класса.

Примечание — В OntoML-языке допускается представлять полное множество ограничительных условий, определенных в информационной модели ИСО 13584-42:2010.

Каждый тип данных определяется как подтип комплексного XML-типа данных **CONSTRAINT_Type**. Общая структура ограничительных условий иллюстрируется рисунком 84.

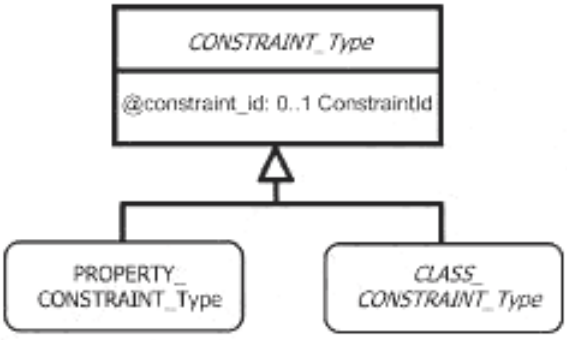


Рисунок 84 — Общая структура ограничительных условий

Определение внутреннего элемента:

Элемент **@constraint_id:** Определяет возможное значение **ConstraintId**, которое идентифицирует ограничительное условие.

Определения внешних типов:

Тип **ConstraintId:** См. 9.1.4.

Тип **CLASS_CONSTRAINT_Type:** Ограничительное условие, связанное с экземплярами класса, см. 8.5.1.

Тип **PROPERTY_CONSTRAINT_Type:** Ограничительное условие, связанное со значением свойства, см. 8.5.3.

Перечень ограничительных условий

Ограничительное условие, определенное с помощью XML-элемента **constraint_id,** является уникальным в экземпляре OntoML-документа.

8.5.1 Ограничительное условие на ссылку

В зависимости от используемого контекста ограничительное условие может быть либо ссылочным, либо явно определенным. Структура класса ограничительного условия на ссылку приведена на рисунке 85.

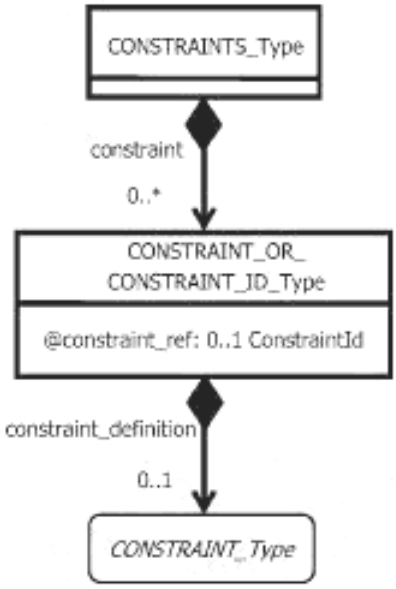


Рисунок 85 — Структура ограничительного условия на ссылку

Определения внутренних элементов:

Элемент **constraint:** Определяет условие, которое ограничивает объектную область значений свойств класса до подкласса его наследуемой области значений.

Элемент **constraint/@constraint_ref:** Определяет идентификатор ограничительного условия.

Элемент **constraint/constraint_definition:** Устанавливает определение ограничительного условия.

Определения внутренних типов:

Тип **CONSTRAINT_OR_CONSTRAINT_ID_Type:** Является спецификацией ограничительного условия, определенного либо в явной форме, либо путем ссылки на идентификатор ограничительного условия.

Тип **CONSTRAINTS_Type:** Является описанием группы ограничительных условий, приведенных либо в явной форме, либо путем ссылки на идентификатор ограничительного условия.

Определения внешних типов:

Тип **ConstraintId:** См. 9.1.

Тип **CONSTRAINT_Type:** См. 8.5.

8.5.2 Ограничительное условие на класс

Ограничительное условие на класс — это условие, которое ограничивает допустимое множество экземпляров класса путем ограничения некоторых свойств или путем определения глобальных ограничительных условий. Структура ограничительных условий на класс приведена на рисунке 86.

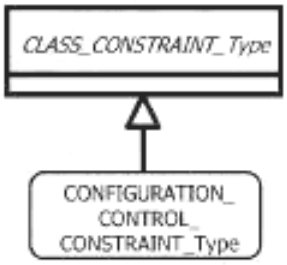


Рисунок 86 — Структура ограничительного условия на класс

Определение внутреннего типа:

Тип **CONFIGURATION_CONTROL_CONSTRAINT_Type:** Является ссылочным условием на экземпляры класса, см. 8.5.2.1.

8.5.2.1 Ограничительные условия на управление конфигурацией

Ограничительные условия на управление конфигурацией позволяют ограничивать множество экземпляров класса, называемых «ссылочными экземплярами класса», конкретным экземпляром, называемым «сопоставительным экземпляром класса», и прямо или косвенно давать ссылки с помощью последовательности свойств. Сопоставительный экземпляр класса — это любой экземпляр, который обеспечивает ссылку на ограничительное условие на управление конфигурацией с помощью XML-элемента **constraints** класса. Это условие определяет дополнительное предварительное условие, которое устанавливает условие на сопоставительный экземпляр класса для ограничения его применения, а также пост-условие, которое определяет ряд значений для некоторых свойств для класса ссылочных экземпляров. Они представляются с помощью комплексного XML-типа данных **CONFIGURATION_CONTROL_CONSTRAINT_Type** (см. рисунок 87).

Примечание 1 — Как предварительное условие, так и постусловие могут ограничивать лишь те свойства, чьи значения в области определены как перечень строковых кодов (см. 8.3.4) или как перечень целочисленных кодов (см. 8.3.8). Указанные свойства могут присваиваться значению либо на уровне экземпляра класса, либо на уровне класса, если они являются как значимыми для класса свойствами, т. е. на которые дается ссылка в XML-элементе **sub_class_properties** класса элементов (см. раздел 6.7.2.1).

Примечание 2 — Свойства, на которые дается ссылка в предварительном условии, применимы для класса, который ссылается на ограничительное условие управления конфигурацией.

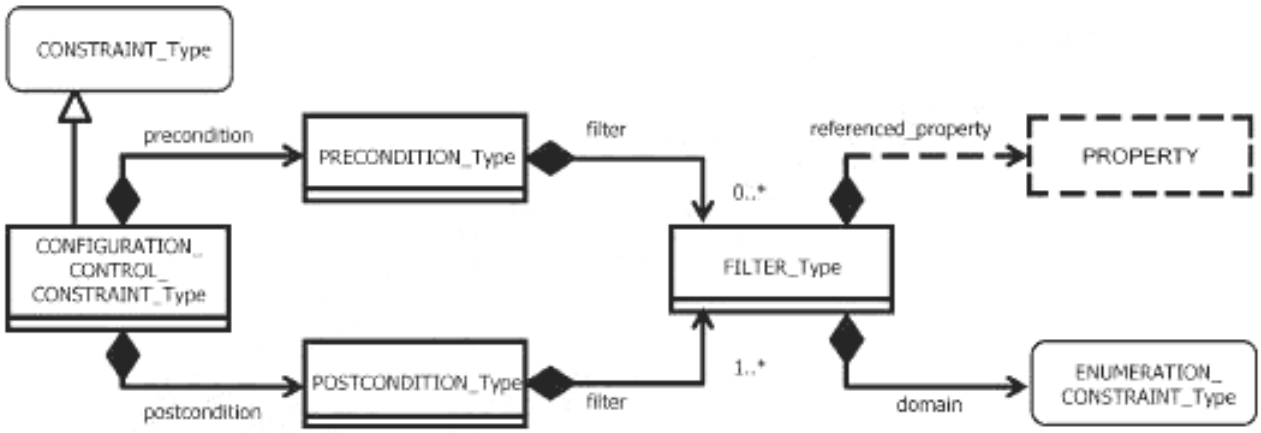


Рисунок 87 — Структура ограничительного условия на управление конфигурацией

Определения внутренних элементов:

Элемент **domain (FILTER_Type):** Определяет ограничительное условие перечислимого типа, которое ограничивает область значений ссылочного свойства.

Элемент **filter (PRECONDITION_Type):** Определяет ограничительные условия, которые должны быть удовлетворены для выполнения ограничений на контроль применимой конфигурации.

Элемент **filter (POSTCONDITION_Type):** Определяет ограничительные условия, которые должны быть удовлетворены для ссылочного экземпляра класса, чтобы он был доступен для ссылки.

Элемент **postcondition:** Определяет описание фильтров, которые должны храниться в ссылочном экземпляре класса, чтобы они были доступны для ссылки.

Элемент **precondition:** Определяет описание фильтров, которые должны храниться в ссылочном экземпляре класса для установления ограничений.

Примечание 3 — Если множество фильтров является пустым, то ограничительное условие будет применимо к любому ссылочному экземпляру класса.

Элемент **referenced_property (FILTER_Type):** Определяет ссылку на свойство, чья область значений ограничивается связанным с ним фильтром.

Определения внутренних типов:

Тип **FILTER_TYPE:** Определяет ограничительные условия на допустимую область свойства, чьим типом данных является перечень строковых кодов (см. 8.3.4) или перечень целочисленных кодов (см. 8.3.8).

Тип **PRECONDITION_Type:** Определяет условия, накладываемые на ссылочный экземпляр класса для ограничения его применения.

Тип **POSTCONDITION_Type:** Определяет допустимое множество значений для некоторых свойств ссылочного класса экземпляров.

Определения внешних типов:

Тип **CONSTRAINT_Type:** См. 8.5.

Тип **ENUMERATION_CONSTRAINT_Type:** Определяет групповые ограничительные условия, см. 8.5.3.3.6.

Перечень ограничительных условий:

Основным типом данных свойства, на который дается ссылка с помощью элемента **referenced_property,** должен быть либо тип **NON_QUANTITATIVE_INT_TYPE_Type,** либо тип **NON_QUANTITATIVE_CODE_TYPE_Type**.

Элемент **domain** должен определять ограничительное условие, которое совместимо с исходной областью значений свойства, на которое ссылка дается с помощью XML-элемента **referenced_property**.

8.5.3 Ограничительное условие на свойство

Ограничительное условие на свойство — это условие, которое ограничивает допустимое множество экземпляров класса единственной областью значений одного из его свойств. Структура ограничительных условий на свойство приведена на рисунке 88.

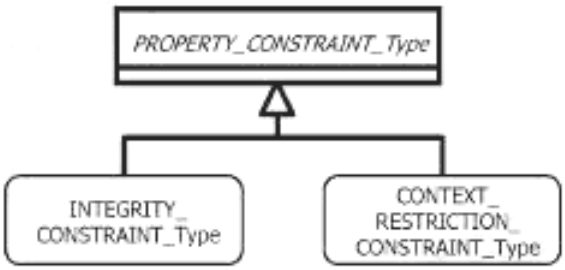


Рисунок 88 — Структура ограничительного условия на свойство

Определения внешних типов:

Тип **CONTEXT_RESTRICTION_CONSTRAINT_Type:** Определяет ограничительные условия на состояния свойства, см. 8.5.3.1.

Тип **INTEGRITY_CONSTRAINT_Type:** Определяет групповые ограничительные условия, см. 8.5.3.2.

8.5.3.1 Ограничительное условие на контекст

Ограничительное условие на контекст применимо к свойствам, которые являются контекстно-зависимыми.

Примечание 1 — Контекстно-зависимые свойства определены в 6.7.4.

Это условие определяет, что область значений контекстного параметра, используемого для определения значения этого контекстно-зависимого свойства, ограничивается использованием любого вида существующего ограничительного условия на область значений.

Примечание 2 — Контекстные параметры определены в 6.7.4.

Это условие определяется с помощью комплексного XML-типа данных **INTEGRITY_CONSTRAINT_Type** (см. рисунок 89).

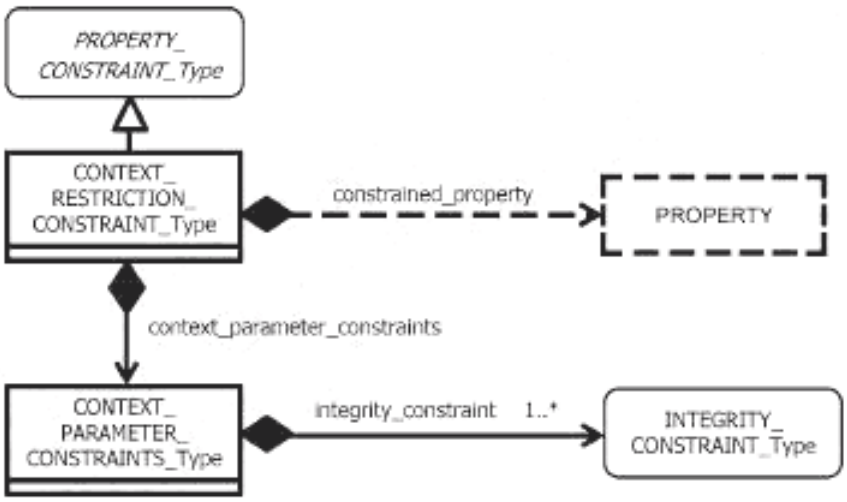


Рисунок 89 — Структура ограничительного условия на контекст

Определения внутренних элементов:

Элемент **constrained_property:** Определяет свойство, для которого применимо ограничительное условие.

Примечание 3 — Ссылочное свойство является контекстно-зависимым (см. раздел 6.7.4).

Элемент **context_parameter_constraints**: Определяет множество ограничительных условий, которые применимы к области значений каждого контекстного параметра ссылочного свойства.

Элемент **integrity_constraint (CONTEXT_PARAMETER_CONSTRAINTS_Type):** Определяет ограничительное условие на целостность, которое ограничивает допустимую область контекстного параметра ограничиваемого контекстно-зависимого свойства.

Пример — Свойство resistance of a thermistor (комплексный XML-тип данных DEPENDENT_P_DET_Type) зависит от свойства ambient temperature (комплексный XML-тип CONDITION_DET_Type), чьей областью значений является тип INT_TYPE_Type. Ограничительное условие на тип данных CONTEXT_RESTRICTION_CONSTRAINT_Type содержит требования, чтобы свойство ambient temperature принимало значение 25° путем применения ограничительного условия на диапазон значений (комплексный XML-тип данных RANGE_CONSTRAINT_Type).

Определение внутреннего типа:

Тип **CONTEXT_PARAMETER_CONSTRAINTS_Type:** Является описанием множества ограничений, которые применимы к области значений контекстных параметров.

Определения внешних типов:

Тип **PROPERTY_CONSTRAINT_Type:** См. 8.5.3.

Тип **INTEGRITY_CONSTRAINT_Type:** Является ограничительным условием на область значений свойства, см. 8.5.3.2.

Перечень ограничительных условий:

Свойство, ссылка на которое дается с помощью XML-элемента **constrained_property,** должно иметь основной тип данных **DEPENDENT_P_DET_Type**.

Множество свойств, чья область значений ограничивается элементом **context_parameter_constraints,** должны быть контекстными параметрами, от которых зависит свойство, ссылка на которое дается с помощью XML-элемента **constrained_property**.

8.5.3.2 Ограничительное условие на целостность данных

Ограничительное условие на целостность данных позволяет четко выражать, что для некоторого свойства конкретного класса (как результат определения класса и всех его подклассов) допускается только ограничение области значений, определенное с помощью типа данных. Данное ограничение представляется с помощью комплексного XML-типа данных **INTEGRITY_CONSTRAINT_Type** (см. рисунок 90).

Пример — В справочном словаре, определенном в ИСО 13584-511 для крепежа, элемент metric threaded bolt/screw имеет свойство head properties, которое может иметь в качестве значения элемент любого подкласса класса признаков head. Если элемент metric threaded bolt/screw также является элементом подкласса hexagon head screw, то элементом head properties может быть только элемент hexagon head класса признаков; кроме того, элемент metric threaded bolt/screw не может быть элементом подкласса hexagon head screw.

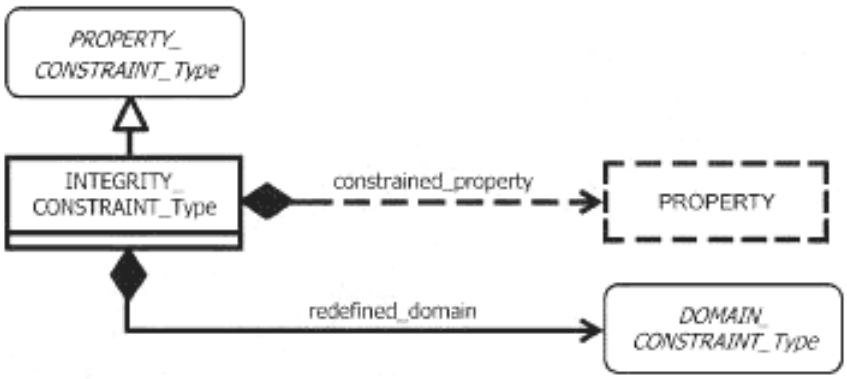


Рисунок 90 — Структура ограничительного условия на целостность данных

Определения внутренних элементов:

Элемент **constrained_property:** Определяет свойство, для которого применимо ограничительное условие.

Элемент **redefined_domain:** Определяет ограничительное условие на целостность данных, которое применимо к области значений ограничиваемого свойства.

Определения внешних типов:

Тип **DOMAIN_CONSTRAINT_Type:** Является конкретным ограничительным условием, которое применимо к ссылочному свойству, см. 8.5.3.3.

Тип **PROPERTY_CONSTRAINT_Type:** См. 8.5.3.

Перечень ограничительных условий:

Элемент **redefined_domain** должен определять ограничение, которое совместимо с исходной областью значений свойства и ссылка на которое дается с помощью XML-элемента **constrained_property**.

8.5.3.3 Ограничительное условие на область значений

OntoML-язык предоставляет ресурсы для описания различных видов ограничительных условий на области значений. Каждое ограничительное условие определяется как подтип комплексного XML-типа **DOMAIN_CONSTRAINT_Type**. На рисунке 91 приведено общее представление о допустимых ограничительных условиях на область значений свойств.

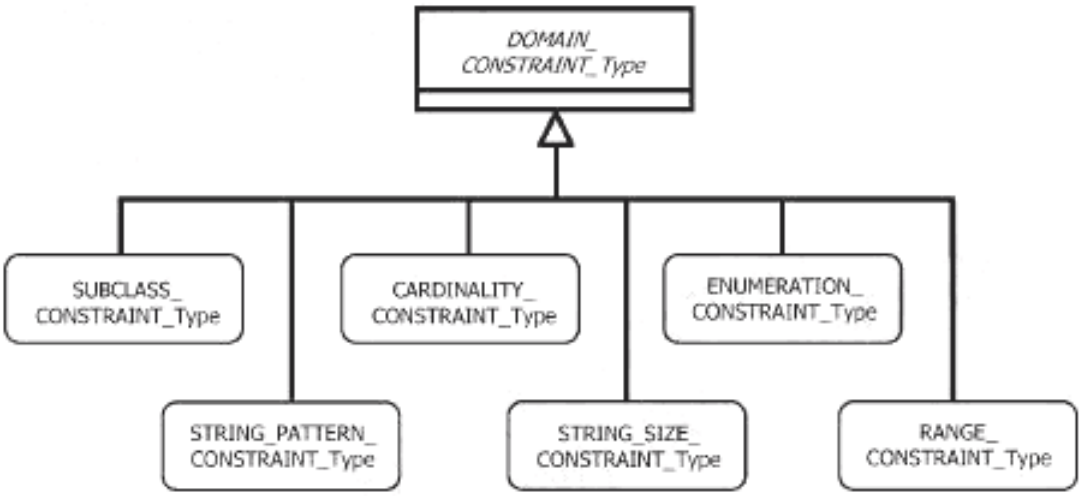


Рисунок 91 — Структура ограничительного условия на область значений

Определения внешних типов:

Тип **CARDINALITY_CONSTRAINT_Type:** Определяет число ограничительных условий, см. 8.5.3.3.3.

Тип **RANGE_CONSTRAINT_Type:** Определяет перечень диапазонов значений для ограничительных условий, см. 8.5.3.3.5.

Тип **STRING_PATTERN_CONSTRAINT_Type:** Определяет перечень характеристик ограничительных условий, см. 8.5.3.3.2.

Тип **STRING_SIZE_CONSTRAINT_Type:** Определяет перечень размеров строк ограничительных условий, см. 8.5.3.3.4.

Тип **SUBCLASS_CONSTRAINT_Type:** Определяет перечень подклассов ограничительных условий, см. 8.5.3.3.1.

Тип **ENUMERATION_CONSTRAINT_Type:** Определяет перечень ограничительных условий, см. 8.5.3.3.6.

8.5.3.3.1 Ограничительное условие на подкласс

Ограничительное условие на подкласс применимо к свойствам, чьи области значений определяются с помощью типа экземпляра класса (см. 8.3.10), указывают на то, что область значений для свойства ограничена подклассом данного класса и представляется комплексным XML-типом данных **SUBCLASS_CONSTRAINT_Type** (см. рисунок 92).

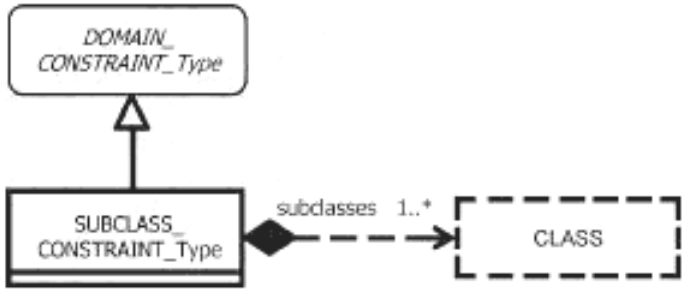


Рисунок 92 — Структура ограничительного условия на подкласс

Определение внутреннего элемента:

Элемент **subclasses:** Определяет ссылки на онтологические понятия класса, которые дополнительно определяют новую область значений ограничиваемого информационного объекта.

Пример — Рассмотрим свойство (комплексного XML-типа данных NON_DEPENDENT_P_DET_Type), называемое screw_head и определенное в контексте класса screw (комплексного XML-типа ITEM_CLASS_Type), чьей областью значений (комплексного XML-типа данных CLASS_REFERENCE_TYPE_Type) является класс head (комплексного XML-типа данных ITEM_CLASS_Type), который определяет основные характеристики любого вида головок винтов. Кроме того, рассмотрим класс hexagonal_screw (комплексного XML-типа данных ITEM_CLASS_Type), подкласс класса screw и класс hexagonal_head (комплексного XML-типа данных ITEM_CLASS_Type), подкласс класса head. В классе hexagonal_screw область значений для элемента screw_head может быть ограничена подклассом класса head, т. е. классом hexagonal_head. Указанное ограничение будет выражаться путем определения специального ограничительного условия для типа данных SUBCLASS_CONSTRAINT_Type, чье значение XML-элемента будет ссылкой на конкретный тип данных ITEM_CLASS_Type, представляющий класс hexagonal_head class.

Определение внешнего типа:

Тип **DOMAIN_CONSTRAINT_Type.** См. 8.5.3.3.

8.5.3.3.2 Ограничительное условие на характеристики строки

Ограничительное условие на характеристики строки применимо к свойствам, чья область значений определяется с помощью строки.

Примечание 1 — Значением свойства строки является либо тип данных **STRING_TYPE_Type** (см. 8.3.2), либо тип данных **NON_TRANSLATABLE_STRING_TYPE_Type** (см. 8.3.2), либо тип данных **TRANSLATABLE_STRING_TYPE_Type** (см. 8.3.2), либо тип данных **URI_TYPE_Type** (см. 8.3.2), либо тип данных **NON_QUANTITATIVE_CODE_TYPE_Type** (см. 8.3.4), либо тип данных **DATE_DATA_TYPE_Type** (см. 8.3.3), либо тип данных **TIME_DATA_TYPE_Type** (см. 8.3.3), либо тип данных **DATE_TIME_DATA_TYPE_Type** (см. 8.3.3).

Ограничительное условие на характеристики строки определяет, что область значений свойства ограничивается в соответствии со значениями строки, которые согласованы с конкретным шаблоном характеристик. Он представляется с помощью комплексного XML-типа данных **STRING_PATTERN_CONSTRAINT_Type** (см рисунок 93).

Для свойств, чей тип данных определен как тип **STRING_TYPE_Type,** тип **NON_TRANSLATABLE_STRING_TYPE_Type,** тип **URI_TYPE_Type,** тип **DATE_DATA_TYPE_Type,** тип **TIME_DATA_TYPE_Type** или тип **DATE_TIME_DATA_TYPE_Type,** ограничительное условие применимо к (уникальной) строке, которая является значением типа данных.

Для свойств, чей тип данных определен как тип **TRANSLATED_STRING_TYPE_Type,** ограничительное условие применимо к строке, которая представлена на языке оригинала, на котором область значений свойства была определена. Этот язык оригинала может быть определен в XML-элементе **source_language** комплексного XML-типа данных **CLASS_TYPE** (см. раздел 6.7.2.1), или комплексного XML-типа данных **PROPERTY_Type** (см. 6.7.4), или комплексного XML-типа данных **DATATYPE_Type** (см. 6.7.6), или комплексного XML-типа данных **DOCUMENT_Type** (см. раздел 6.7.7). Если данный атрибут не существует, то этот язык оригинала будет считаться известным пользователю словаря.

Для свойств, чей тип данных определен как тип **NON_QUANTITATIVE_CODE_TYPE_type,** ограничительное условие применимо только к коду.

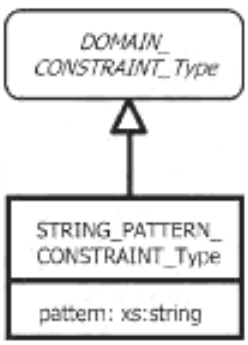


Рисунок 93 — Структура ограничительного условия на характеристики строки

Определение внутреннего элемента:

Элемент **pattern:** Определяет характеристики строчных значений, которые допускаются в качестве значений для свойства, идентифицированного с помощью ограниченного свойства.

Примечание 2 — Синтаксис значения XML-элемента **pattern** основывается на регулярном выражении и связанными с ним сопоставительными алгоритмами, определенными с помощью Части 2: Рекомендации по выбору типов данных XML-диаграммы.

Пример — Характеристика XML-диаграммы, которая соответствует SQL SIMILAR-выражению "[0-9][0-9][0-9][0-9]-[0-9][0-9]-[0-9][0-9]", является характеристикой "[0-9]{4}\-[0-9]{2}\-[0-9]{2}", что позволяет согласовывать строки в виде "2010-03-24".

Определение внешнего типа:

Тип **DOMAIN_CONSTRAINT_Type:** См. 8.5.3.3.

8.5.3.3.3 Ограничительное условие на количество элементов

Элемент **cardinality_constraint** ограничивает количество элементов в групповом типе данных.

Примечание 1 — Окончательный диапазон количества элементов — это область пересечения ранее существовавших диапазонов количеств элементов и одного диапазона, определяемого с помощью соответствующего ограничительного условия.

Примечание 2 — Ограничительные условия на количество элементов не допускаются для массивов (см. 8.3.9.4).

Это условие представляется с помощью комплексного XML-типа данных **CARDINALITY_CONSTRAINT_Type** (см. рисунок 94).

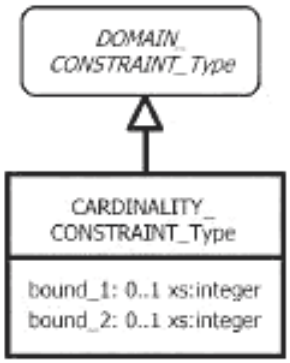


Рисунок 94 — Структура ограничительного условия на количество элементов

Определения внутренних элементов:

Элемент **bound_1:** Определяет нижнюю границу количества элементов.

Элемент **bound_2:** Определяет верхнюю границу количества элементов.

Определение внешнего типа:

Тип **DOMAIN_CONSTRAINT_Type:** См. раздел 8.5.3.3.

Перечень ограничительных условий:

Если элемент **bound_1** не существует, то минимальное количество элементов равно нулю.

Если элемент **bound_2** не существует, то ограничительное условие на количество элементов будет отсутствовать.

Если элемент **bound_1** существует, то его значение должно быть больше или равно нулю.

Если предоставляются оба элемента **bound_1** и **bound_2**, то элемент **bound_2** должен быть больше или равен элементу **bound_1**.

8.5.3.3.4 Ограничительное условие на размер строки

Ограничительное условие на размер строки применяется к свойству, чья область значений определяется как строка.

Примечание 1 — Область значений свойства в виде строки — это либо строка (см. раздел 8.3.2), либо непереводимая строка (см. 8.3.2), либо переводимая строка (см. раздел 8.3.2), либо удаленный http-адрес (см. 8.3.2), либо перечень строковых кодов (см. раздел 8.3.3).

Ограничительное условие на размер строки определяет, что область значений для свойства ограничивается, возможно, минимальной и/или максимальной длиной и представляется с помощью комплексного XML-типа данных **STRING_SIZE_CONSTRAINT_Type** (см. рисунок 95).

Примечание 2 — Для свойств, чей тип данных определен как тип **TRANSLATED_STRING_TYPE_Type**, ограничительное условие будет применимо к любому языковому представлению строки.

Примечание 3 — Для свойств, чей тип данных определен как тип **NON_QUANTITATIVE_CODE_TYPE_type**, ограничительное условие будет применимо к коду.

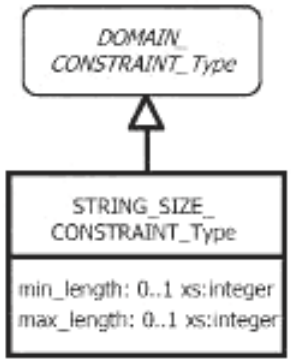


Рисунок 95 — Структура ограничительного условия на размер строки

Определения внутренних элементов:

Элемент **max_length:** Определяет максимальную длину строк, которые допускаются в качестве значения для ограниченного свойства.

Элемент **min_length:** Определяет минимальную длину строк, которые допускаются в качестве значения для ограниченного свойства.

Примечание 4 — Значение элемента **min_length** больше нуля и меньше или равно значению элемента **max_value**.

Пример — Свойство, чьей областью значений является строка (комплексного XML-типа данных STRING_TYPE_Type) и которое определено в заданном классе, может быть ограничено в подклассе данного класса, поскольку оно будет иметь более 10 символов за счет ограничительного условия на тип данных STRING_SIZE_CONSTRAINT_Type и присвоения значения 10 XML-элементу max_length.

Определение внешнего типа:

Тип **DOMAIN_CONSTRAINT_Type:** См. 8.5.3.3.

Перечень ограничительных условий:

Если элемент **min_length** не существует, то минимальная длина строки равна 0.

Если элемент **max_length** не существует, то ограничительное условие на длину строки будет отсутствовать.

Если элемент **min_length** существует, то значение длины строки должно быть больше или равно нулю.

Если предоставляются оба элемента **min_length** и **max_length**, то элемент **max_length** должен быть больше или равен элементу **min_length**.

8.5.3.3.5 Ограничительное условие на диапазон значений

Ограничительное условие на диапазон значений применимо к свойствам, чьи области значений определяются номером.

Пример 1 — Определяемая номером область значений — это данные типа «число» (см. 8.3.5), «целое число» (см. 8.3.5), «целое значение валюты» (см. 8.3.6), «целая мера» (см. 8.3.7), «действительное число» (см. 8.3.5), «действительное значение валюты» (см. 8.3.6), «действительная мера» (см. 8.3.7) или «перечень целочисленных кодов» (см. 8.3.8).

Ограничительное условие на диапазон значений определяет, что область значений для свойства ограничивается подмножеством ее значений, определяемых диапазоном и представляемых с помощью комплексного XML-типа данных **RANGE_CONSTRAINT_Type** (см. рисунок 96).

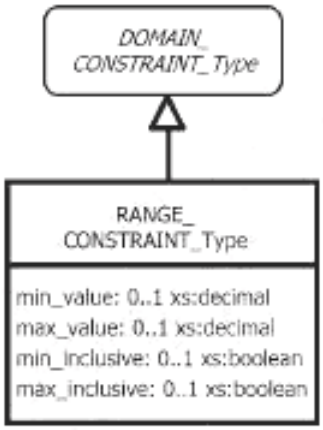


Рисунок 96 — Структура ограничительного условия на диапазон значений

Определение внутреннего элемента:

Элемент **max_value:** Определяет число, характеризующее верхнюю границу диапазона значений.

Элемент **max_inclusive:** Если значение истинно, то устанавливается, что элемент **max_value** входит в заданный диапазон.

Элемент **min_value:** Определяет число, характеризующее нижнюю границу диапазона значений.

Элемент **min_inclusive:** Если значение истинно, то устанавливается, что элемент **min_value** входит в заданный диапазон.

Пример 2 — Свойство, чьей областью значений является целое число (комплексного XML-типа данных INT_TYPE_Type) и которое определено в заданном классе, может быть ограничено в подклассе данного класса как диапазон [10..50] путем определения ограничительного условия для типа данных RANGE_CONSTRAINT_Type и присвоения значений 10 — XML-элементу min_value и значения 50 — XML-элементу max_value.

Определение внешнего типа:

Тип **DOMAIN_CONSTRAINT_Type:** См. 8.5.3.3.

Перечень ограничительных условий:

Элемент **min_value** должен быть меньше или равен элементу **max_value**.

Элементы **min_value** и **max_value** оба должны быть целыми числами или действительными числами.

Либо элемент **min_value,** либо элемент **max_value** должны быть определены.

Если элемент **min_inclusive** не определен, то не должно быть никаких ограничений на нижнюю границу задаваемого диапазона значений.

Если элемент **max_inclusive** не определен, то не должно быть никаких ограничений на верхнюю границу задаваемого диапазона значений.

Если элемент **min_value** не определен, то и элемент **min_inclusive** не должен быть определен.

Если элемент **max_value** не определен, то и элемент **max_inclusive** не должен быть определен.

Если элемент **min_value** определен, то и элемент **min_inclusive** должен быть определен.

Если элемент **max_value** определен, то и элемент **max_inclusive** должен быть определен.

8.5.3.3.6 Ограничительное условие на перечень

Ограничительное условие на перечень позволяет ограничивать область значений типов данных списком, определенным в расширении. Порядок, определенный с помощью этого списка, является рекомендуемым для представления. Конкретное описание может дополнительно связываться с каждым значением из списка с помощью типа данных **NON_QUANTITATIVE_INT_TYPE_Type,** в котором i-ое значение описывает содержание i-ого значения в списке.

Данное ограничительное условие на перечень представляется с помощью комплексного XML-типа данных **ENUMERATION_CONSTRAINT_Type** (см. рисунок 97).

Примечание 1 — Для типа данных «валюта» (см. 8.3.6) или «мера» (см. 8.3.7), связанного с альтернативной единицей, ограничительное условие применимо для любого из этих типов данных.

Примечание 2 — Для типа данных «переводимые строки» (см. 8.3.2) ограничительное условие применимо для любого специального языкового представления строки.

Примечание 3 — Если к свойству, уже связанному с ограничительным условием на перечень, применяют другое подобное ограничительное условие в некотором суперклассе, то должны применяться оба эти ограничения. Следовательно, допустимый набор значений будет определяться областью пересечения обоих подмножеств.

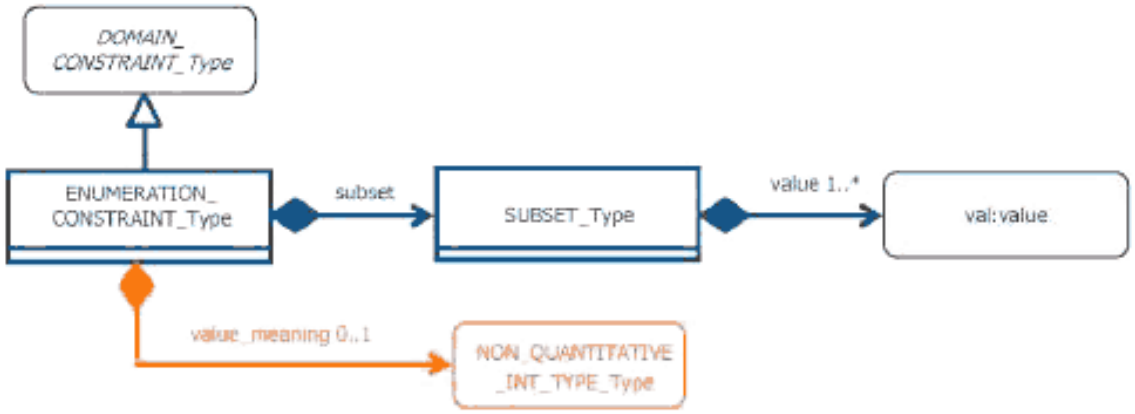


Рисунок 97 — Структура ограничительного условия на перечень

Пример 1 — Предположим, что свойство, чьим идентификатором является 0123-ABCD#02-P1#1, имеет областью значений целое число. В контексте класса, идентифицированного с помощью идентификатора 0123-ABCD#01-C1#1 IRDI, это свойство связывается с ограничительным условием на перечень, где допустимые значения определяются с помощью следующего подмножества целых чисел {1, 3, 5, 7}. После этого в классе, идентифицированном с помощью идентификатора 0123-ABCD#01-C1#1 IRDI и любого из его подклассов, идентифицированное с помощью идентификатора 0123-ABCD#02-P1#1 IRDI свойство может принимать только одно из следующих значений: 1 или 3 или 5 или 7. OntoML-представление данного ограничительного условия на перечень будет принимать следующий вид:

```
<ontoml:class xsi:type="ontoml:ITEM_CLASS_Type" id="0123-ABCD#01-C1#1"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:ontoml="urn:iso:std:iso:13584:-32:ed-1:tech:xml-schema:ontoml"
    xmlns:val="urn:iso:std:iso:ts:29002:-10:ed-1:tech:xml-schema:value">
    ...
    <constraints>
        <constraint>
            <constraint_definition xsi:type="ontoml:INTEGRITY_CONSTRAINT_Type">
                <constrained_property property_ref="0123-ABCD#02-P1#1"/>
                <redefined_domain xsi:type="ontoml:ENUMERATION_CONSTRAINT_Type">
                    <subset>
                        <val:integer_value>1</val:integer_value>
                        <val:integer_value>3</val:integer_value>
                        <val:integer_value>5</val:integer_value>
                        <val:integer_value>7</val:integer_value>
                    </subset>

                </redefined_domain>
            </constraint_definition>
        </constraint>
    </constraints>
</ontoml:class>
```

Пример 2 — Предположим, что свойство, чьим идентификатором является 0123-ABCD#02-P1#1, имеет область значений в виде списка из минимум 1 и максимум 4 целых чисел. В контексте класса, идентифицированного с помощью идентификатора 0123-ABCD#01-C1#1 IRDI, это свойство связывается с ограничительным условием на перечень с помощью следующего списка целых чисел { {1} , {3, 5}, {7}, {1, 3, 7} }. В этом контексте это будет означать, что с помощью идентификатора 0123-ABCD#02-P1#1 свойство может принимать следующие значения: {1} или {3, 5} или {7} или {1, 3, 7}. OntoML-представление данного ограничительного условия на перечень будет принимать следующий вид:

```
<ontoml:class xsi:type="ontoml:ITEM_CLASS_Type" id="0123-ABCD#01-C1#1"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:ontoml="urn:iso:std:iso:13584:-32:ed-1:tech:xml-schema:ontoml"
    xmlns:val="urn:iso:std:iso:ts:29002:-10:ed-1:tech:xml-schema:value">
    ...
    <constraints>
        <constraint>
            <constraint_definition xsi:type="ontoml:INTEGRITY_CONSTRAINT_Type">
                <constrained_property property_ref="0123-ABCD#02-P1#1"/>
                <redefined_domain xsi:type="ontoml:ENUMERATION_CONSTRAINT_Type">
                    <subset>
                        <val:sequence_value>
                            <val:integer_value>1</val:integer_value>
                        </val:sequence_value>
                        <val:sequence_value>
                            <val:integer_value>3</val:integer_value>
                            <val:integer_value>5</val:integer_value>
                        </val:sequence_value>
                        <val:sequence_value>
                            <val:integer_value>7</val:integer_value>
                        </val:sequence_value>
                        <val:sequence_value>
                            <val:integer_value>1</val:integer_value>
                            <val:integer_value>3</val:integer_value>
                            <val:integer_value>7</val:integer_value>
                        </val:sequence_value>
                    </subset>
                </redefined_domain>
            </constraint_definition>
        </constraint>
    </constraints>
</ontoml:class>
```

Пример 3 — Предположим, что свойство (определенное в классе, идентифицированном с помощью идентификатора 0123-ABCD#01-C1#1 IRDI), чьим идентификатором является 0123-ABCD#02-P1#1 и чьей областью значений является действительная мера, выражаемая в миллиметрах, ограничивается следующим множеством значений: {10.5, 30}. Кроме того, каждое значение ограниченного множества связывается с конкретным содержанием, выраженным на французском и английском языках, т.е. 10.5 соответствует выражению "a little value" (на английском) или "une petite valeur" (на французском); 30 соответствует выражению "big value" (на английском) или "une grande valeur" (на французском). В OntoML-языке эта область значений свойства будет представляться как комплексный XML-тип данных REAL_MEASURE_TYPE_Type (см. раздел 8.3.7), ограниченный путем определения ограничительного условия на тип данных ENUMERATION_CONSTRAINT_Type. Это условие определяет два возможных значения: 10.5 и 30. Кроме того, оба значения связываются с конкретным содержанием (XML-элементом value_meaning), которое выражается с использованием перечня (комплексного XML-типа данных NON_QUANTITATIVE_INT_Type). Каждый элемент этого перечня (XML-элемент dic_value):

— во-первых, связывается с кодом значения (целым числом, представленным с помощью комплексного XML-типа данных INT_DIC_VALUE_Type), идентифицирующим позицию значения, с которым связано текущее содержание (10.5 – это первое значение множества значений, затем связываемое с кодом значения, равным 1; 30 – это второе значение этого множества, затем связываемое с кодом значения, равным 2);

— первое значение множества значений затем связывается с кодом значения, равым 1; 30 – это второе значение множества значений, которое затем связывается с кодом значения, равным 2);

— во-вторых, связывается (посредством XML-элемента preferred_name) с его содержанием (переведенной меткой, см. раздел 8.1).

OntoML-представление вышеприведенного примера имеет следующий вид:

```
<ontoml:property xsi:type="ontoml:NON_DEPENDENT_P_DET_Type" id="0123-ABCD#02-PROPERTY#1"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:ontoml="urn:iso:std:iso:13584:-32:ed-1:tech:xml-schema:ontoml"
    xmlns:val="urn:iso:std:iso:ts:29002:-10:ed-1:tech:xml-schema:value">
    ...
    <domain xsi:type="ontoml:REAL_MEASURE_TYPE_Type">
        <constraints>
            <constraint xsi:type="ontoml:ENUMERATION_CONSTRAINT_Type">
                <subset>
                    <val:real_value>10.5</val:real_value>
                    <val:real_value>30</val:real_value>
                </subset>
                <value_meaning>
                    <its_values>
                        <dic_value xsi:type="ontoml:INT_DIC_VALUE_Type">
                            <preferred_name>
                                <label language_code="en">a little value</label>
                                <label language_code="fr">une petite valeur</label>
                            </preferred_name>
                            <value_code>1</value_code>
                        </dic_value>
                        <dic_value xsi:type="ontoml:INT_DIC_VALUE_Type">
                            <preferred_name>
                                <label language_code="en">a big value</label>
                                <label language_code="fr">une grande valeur</label>
                            </preferred_name>
                            <value_code>2</value_code>
                        </dic_value>
                    </its_values>
                </value_meaning>
            </constraint>
        </constraints>
        <unit>
            <structured_representation xsi:type="ontoml:SI_UNIT_Type">
                <prefix>MILLI</prefix>
                <name>METRE</name>
            </structured_representation>
        </unit>
    </domain>
</ontoml:property>
```

Определения внутренних элементов:

Элемент **subset:** Определяет список, описывающий подмножество значений, которые допускаются как возможные значения для ограниченного свойства.

Примечание 4 — Порядок, определенный с помощью этого списка, является рекомендованным для целей представления.

Элемент **subset.value:** Определяет значения, которые допускаются как возможное значение для ограниченного свойства.

Примечание 5 — Представление значений соответствует приведенным в ИСО/ТС 29002-10.

Элемент **value_meaning:** Определяет дополнительное описание, которое может связываться с каждым значением **subset** посредством перечня целочисленных кодов, чье i-ое значение описывает содержание i-ого значения элемента **subset**.

Определение внутреннего типа:

Тип **SUBSET_Type:** Является множеством значений для типа, определенного с помощью комплексного XML-типа данных **DATA_TYPE_Type**.

Определение внешнего типа:

Элемент **val:value:** Является описанием напечатанного значения.

Примечание 6 — Элемент **val:value** определен в ИСО/ТС 29002-10 как формат обмена данными о продукции.

Тип **DOMAIN_CONSTRAINT_Type:** См. раздел 8.5.3.3.

Тип **NON_QUANTITATIVE_INT_TYPE_Type:** См. раздел 8.3.8.

Перечень ограничительных условий:

Если предоставляется элемент **value_meaning**, то размер группы элементов, представляющей связанные с ним коды значений, должен быть равным размеру группы элементов **subset**.

Если предоставляется элемент **value_meaning**, то каждое значение группы элементов **subset** должно быть связано с конкретным содержанием.

Если предоставляется элемент **value_meaning**, то i-ое значение элемента **dic_value** будет описывать i-ое значение подмножества элементов.

8.6 Апостериорное семантическое соотношение

Апостериорное семантическое соотношение – это ориентированное соотношение между двумя классами, из которого моделируется эквивалентность (или соответствие) свойств, описывающее способ расчета свойств одного из классов по свойствам другого класса. Все соответствия классов обладают одним и тем же положением (ориентацией), которое зависит от семантического соотношения.

В одном и том же *апостериорном* семантическом соотношении может быть определено несколько эквивалентностей свойств. Классы, входящие в одно или несколько *апостериорных* семантических соотношений, могут иметь свойства, которые не являются отображаемыми.

Примечание — Не требуется, чтобы все характеристики экземпляров каждого из классов представлялись свойствами (в словаре) и/или значениями в библиотеке. Могут отображаться только свойства, определенные в онтологии.

Апостериорное семантическое соотношение представляется с помощью абстрактного комплексного XML-типа данных **A_POSTERIORI_SEMANTIC_RELATIONSHIP_Type** (см. рисунок 98).

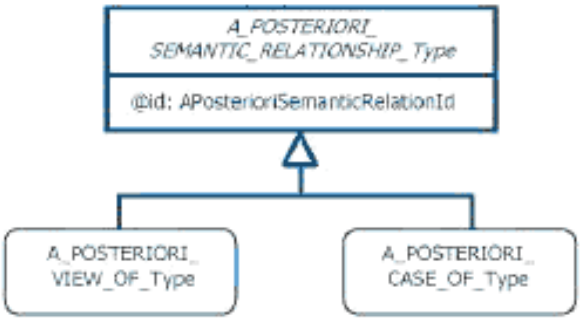


Рисунок 98 — Структура апостериорного семантического соотношения

Конкретное *апостериорное* семантическое соотношение определяется посредством одного из абстрактных комплексных подтипов XML-типа данных **A_POSTERIORI_SEMANTIC_RELATIONSHIP_Type**.

Определение внутреннего элемента:

Элемент **@id:** Определяет идентификатор *апостериорного* семантического соотношения.

Определение внутреннего типа:

Тип **APosterioriSemanticRelationId:** См. раздел 9.1.

Определения внешних типов:

Тип **A_POSTERIORI_CASE_OF_Type:** Является условным (*case-of*) *апостериорным* семантическим соотношением, см. раздел 8.6.1.

Тип **A_POSTERIORI_VIEW_OF_Type:** Является производным (*view-of*) *апостериорным* семантическим соотношением, см. раздел 8.6.2.

8.6.1 *Апостериорное* отображение в условном семантическом отношении

Условное (*case-of*) *апостериорное* семантическое отношение позволяет определять отношение включения между классами, принадлежащими различным словарям-справочникам. Если класс *A* является условием для класса B, то это будет означать, что все экземпляры класса *A* также будут и экземплярами класса *B*. Класс *A* обозначается элементом **case_of_sub**, а класс *B* - элементом **case_of_super**. Все отображения свойств описываются как свойства элемента **case_of_super** класса, определенные в его диапазоне значений и могут рассчитываться по свойствам элемента **case_of_sub** класса, определенным в их области значений (см. рисунок 99).

В OntoML-языке это соотношение представляется с помощью комплексного XML-типа данных **A_POSTERIORI_CASE_OF_Type** (см. рисунок 99).

Примечание 1 — Условное отношение позволяет каждой организации определять свой собственный словарь-справочник при предоставлении данных для их интеграции и обмена с другими организациями.

Пример 1 — Предположим, что элемент case_of_super принадлежит стандартной онтологии и элемент case_of_sub принадлежит онтологии пользователя. Апостериорное условное отношение будет позволять экспортировать локальные данные в соответствии со стандартной онтологией.

Примечание 2 — При эквивалентности двух классов *A* и B, т.е. все экземпляры класса *A* также являются и экземплярами класса *B*, а все экземпляры класса *B* являются также и экземплярами класса *A*. Это может представляться с использованием *апостериорных* семантических отношений со свойством, отображаемым с обратной стороны.

Пример 2 — Предположим, что элемент класса case_of_sub принадлежит стандартной онтологии, и что элемент класса case_of_super принадлежит онтологии пользователя. Апостериорное условное отношение будет позволять импортировать данные, описываемые в соответствии со стандартной онтологией в базе данных пользователя.

Примечание 3 — В данной версии OntoML-языка только доступная отображающая функция является соразмерной, однако для последующей стандартизации предназначен тип данных **MAPPING_FUNCTION_Type**.

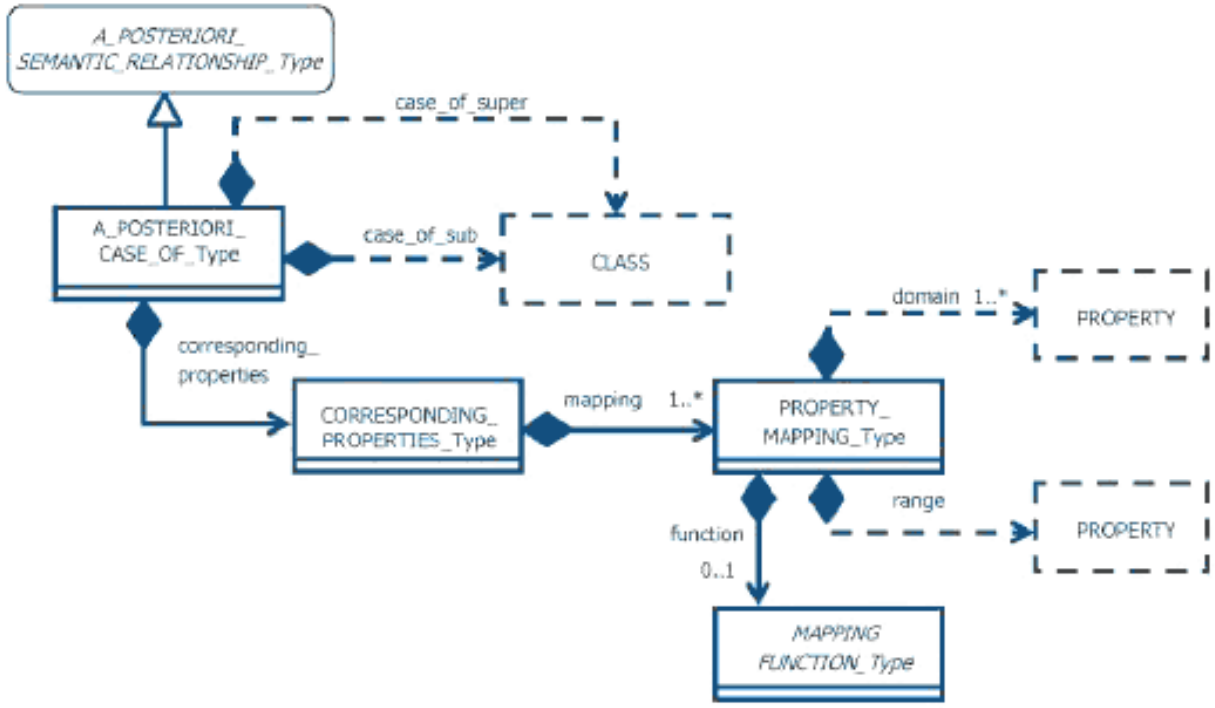


Рисунок 99 — Структура апостериорного условного семантического отношения

Определения внутренних элементов:

Элемент **corresponding_properties:** Определяет множество отображаемых свойств.

Элемент **corresponding_properties/mapping:** Определяет отображение, которое указывает на способ расчета значения (свойства) **range** для всех экземпляров из (свойства) **domain** с помощью соответствующего функционального элемента.

Примечание 4 — Если функциональный XML-элемент не существует, то XML-элемент области значений **domain** будет содержать единственное значение, а отображение указывает на равенство значений свойств **range** и **domain**.

Элемент **corresponding_properties/mapping/domain:** Определяет ссылку на свойства, которые используются для расчета диапазона значений свойства.

Элемент **corresponding_properties/mapping/function:** Определяет дополнительную функцию, которая устанавливает способ расчета значения (свойства) **range** для всех экземпляров из (свойства) **domain**. Если этот элемент не существует, по умолчанию функция принимает значение (свойства) **range**, равное для всех экземпляров из (свойства) **domain**, которые должны быть уникальными.

Элемент **corresponding_properties/mapping/range:** Определяет ссылку на свойство, которое рассчитывается с помощью отображения.

Элемент **case_of_sub:** Определяет ссылку на класс, который является экземпляром элемента **case_of_super** класса отношений.

Примечание 5 — Элементы **case_of_super** и **case_of_sub** ссылочных классов являются элементами классов или элементами классов функциональных моделей.

Элемент **case_of_super:** Определяет ссылку на класс, элемент которого **case_of_sub** является элементом условного класса.

Определения внутренних типов:

Тип **CORRESPONDING_PROPERTIES_Type:** Является хранилищем для множества пар свойств.

Тип **PROPERTY_MAPPING_Type:** Является определением отображения свойства.

Тип **MAPPING_FUNCTION_Type:** Является абстрактным комплексным XML-типом данных, предназначенным для представления отображающей функции и зарезервированным для последующей стандартизации.

Определение внешнего типа:

Тип **A_POSTERIORI_SEMANTIC_RELATIONSHIP_Type:** См. раздел 8.6.

Перечень ограничительных условий:

Классы **Case_of_super** и **case_of_sub,** ссылка на которые приводится с помощью комплексного XML-типа данных **A_POSTERIORI_CASE_OF_Type,** должны быть классами элементов или классами функциональных моделей.

Свойства, ссылка на которые приводится с помощью комплексного XML-типа данных **PROPERTY_MAPPING_Type** с помощью их диапазона значений **range**, должны принадлежать элементу **case_of_super** класса.

Свойства, ссылка на которые дается с помощью комплексного XML-типа данных **PROPERTY_MAPPING_Type** с помощью их области значений **domain**, должны принадлежать элементу **case_of_sub** класса.

Если функция **function**, ссылка на которую дается с помощью комплексного XML-типа данных **PROPERTY_MAPPING_Type**, не существует, то свойство **domain** должно быть уникальным.

8.6.2 Апостериорное отображение в производном семантическом отношении

Производное *(view-of) апостериорное* семантическое отношение позволяет связывать функциональный класс **model**, называемый «моделью», для определения характеристик класса элементов, называемого «элементом». Класс **model** предоставляет дополнительные описательные свойства для каждого элемента с отраслевой точки зрения, определенной с помощью класса функциональных представлений. Каждый экземпляр класса **model** состоит из списка пар «свойства/значение». Подмножество этих свойств, которые входят в XML-элемент **instance_identification,** определяют основные свойства этих экземпляров класса.

Установление соответствия основных свойств со свойствами элементов позволяет идентифицировать, какие экземпляры модели могут согласовываться с каждым элементом экземпляров **item**. Критерий согласования заключается в том, что экземпляр **model** согласуется с экземпляром **item**, если для всех элементов свойства **model**, отображаемых на один элемент свойства **item** (или на их группу) значение свойства экземпляра **model** будет равно результату отображения этого свойства на свойства экземпляра **item**.

Примечание 1 — Если все основные свойства класса функциональных моделей могут отображаться на свойства класса элементов, то каждый элемент связывается по крайней мере с одной функциональной моделью, которая может быть рассчитана для всего класса путем предоставления внешнего соединения.

Пример — Предположим, что изготовитель винтов решает разработать онтологию для описания производимых винтов только одного типа. Они могут описываться с помощью

пяти свойств (признаков): part number, length, thread diameter, euro price и quantity of order, для которых могут быть предложены следующие три онтологические структуры:

1 – Структура Screw class, для которой элемент item_class принадлежит уникальному классу онтологии. Все свойства part number, length, thread diameter представляются как применимые для данного класса свойства. Все технические и коммерческие свойства представляются как применимые для данного класса свойства и объединяются в нем.

2 - Структура Screw class с элементом item_class и screw business class с элементом fm_class_view_of являются двумя классами онтологии. Свойства part number, length и thread diameter представляются как применимые свойства для структуры screw class. Screw business class – это функциональная модель, заявляемая как производный класс screw class, который импортирует свойство part number из класса screw class и заявляет о том, что свойства euro price и quantity of order являются применимыми. Общее свойство part number позволяет устанавливать связь между двумя классами при разделении свойств с жесткими характеристиками и коммерчески-ориентированными свойствами. Технические и коммерческие свойства разделяются на два класса с помощью априорного производного (view-of) соотношения.

3 - Структура Screw class с элементом item_class и класс screw business class с функциональным элементом functional_model_class являются двумя классами, которые могут быть определены в одном и том же классе или в двух раздельных онтологиях. Свойства part number, length и thread diameter представляются как применимые свойства класса screw class. Винт класса business class объявляется с помощью трех применимых свойств: screw_id, euro price и quantity of order. Элемент screw_id содержит номер детали, соответствующий свойствам price и quantity of order. В данном подходе полностью разделяются техническое и коммерческое описания винта. При этом если в некотором контексте будет удобно объединить эти два типа данных, то это может выполняться путем создания апостериорного производного отношения, в котором класс screw business class имеет элемент model, свойство screw class – элемент item, свойство screw_id – элемент range уникального типа PROPERTY_MAPPING_Type, а свойство part number – элемент target.

Примечание 2 — Основные свойства класса функциональных моделей также могут содержать свойства контрольной переменной представления, импортируемые из класса функциональных представлений, ссылка на который дается с помощью класса функциональных моделей. В этом случае значение для свойств контрольной переменной представления должно определяться пользователем для получения каждой конкретной модели, определенной пользователем и соответствующей тому же элементу.

В OntoML-языке производное *апостериорное* семантическое отношение представляется с помощью комплексного XML-типа данных **A_POSTERIORI_VIEW_OF_Type** (см. рисунок 100).

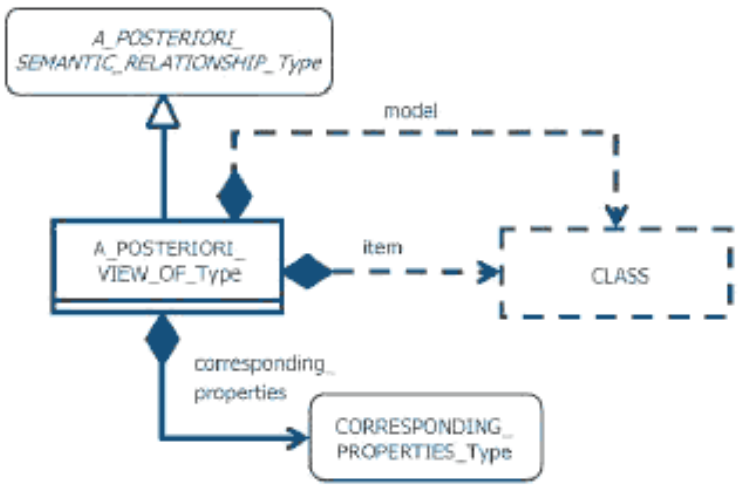


Рисунок 100 — Структура апостериорного производного семантического отношения

Определения внутренних элементов:

Элемент **corresponding_properties:** Определяет множество отображаемых свойств.

Элемент **model:** Определяет ссылку на класс функциональных моделей, которые будут предоставлять дополнительные свойства продукции для класса **item**.

Элемент **item:** Определяет ссылку на класс элементов, для которого классом функциональных моделей является производный (*view-of*) класс.

Определения внешних типов:

Тип **A_POSTERIORI_SEMANTIC_RELATIONSHIP_Type:** См. раздел 8.6.

Тип **CORRESPONDING_PROPERTIES_Type:** Является отображением свойства, см. раздел 8.6.1.

Перечень ограничительных условий:

Класс **model,** ссылка на который дается с помощью комплексного XML-типа данных **A_POSTERIORI_VIEW_OF_Type,** должен быть классом функциональных моделей. Класс **item,** ссылка на который дается с помощью комплексного XML-типа данных **A_POSTERIORI_VIEW_OF_Type,** должен быть классом элементов.

Свойства, ссылка на которые дается с помощью элемента **range** комплексного XML-типа данных **PROPERTY_MAPPING_Type**, должны принадлежать XML-элементу **instance_identification** класса **model**.

Свойства, ссылка на которые дается с помощью элемента **domain** комплексного XML-типа данных **PROPERTY_MAPPING_Type**, должны принадлежать классу **item**.

Если класс **function,** ссылка на который дается с помощью комплексного XML-типа данных **PROPERTY_MAPPING_Type**, не существует, свойство **domain** должно быть уникальным.

8.7 Идентификация спецификации на обмен данными

Спецификация на обмен данными позволяет определять различные характеристики экземпляра OntoML-документа, т.е.:

— подмножества используемых OntoML-спецификаций;

— протоколы возможного обмена представлениями, используемыми при работе с функциональными моделями, которые связаны с классами характеристик;

— метод реализации в OntoML-языке экземпляра OntoML-документа.

Определяются два элемента спецификации на обмен данными, позволяющими определять:

— характеристики обмениваемой онтологии и/или библиотеки (см. раздел 8.7.1).

— характеристики возможных протоколов обмена используемыми представлениями (см. раздел 8.7.2).

8.7.1 Спецификация на обмен онтологическими данными простого уровня: идентификация интегрированной в библиотеку информационной модели

Идентификация интегрированной в библиотеку информационной модели позволяет определять модель, на которой основывается информационный обмен, а также используемый формат.

Примечание 1 — В приложении С определены стандартные значения, которые используются при обмене OntoML-данными. Эти значения также определяются в примечаниях.

Она представляется с помощью комплексного XML-типа данных **LIBRARY_IIM_IDENTIFICATION_Type** (см. рисунок 101).

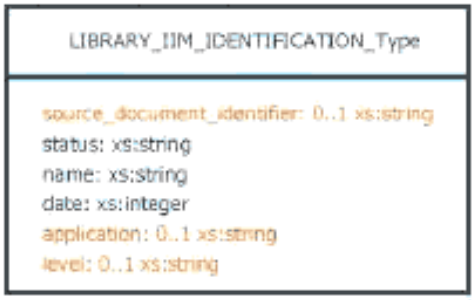


Рисунок 101 — Структура идентификации, интегрированной в библиотеку информационной модели

Определения внутренних элементов:

Элемент **application:** Определяет идентификатор для указания характеристик допустимого функционального подмножества полного определения данных.

Примечание 2 — Значения равны:

1 – при использовании класса соответствия 1 (онтология простого типа), равны 2 при использовании класса соответствия;

2 – (онтология повышенного уровня), равны 3 при использовании класса соответствия;

3 – (библиотека простого уровня), равны 4 при использовании класса соответствия;

4 – (библиотека повышенного уровня).

Элемент **date:** Определяет год, когда спецификация на дату достигает элемента **status.**

Элемент **level:** Определяет идентификатор, который дополнительно характеризует допустимое подмножество элемента **application**.

Примечание 3 — Данный XML-элемент не используется для обмена OntoML-данными.

Элемент **name:** Определяет идентификатор спецификации на данные.

Примечание 4 — В OntoML-языке этим идентификатором является "ONTOML" (заглавные буквы).

Элемент **source_document_identifier:** Определяет идентификатор документа, который содержит спецификацию на данные.

Примечание 5 — Для тех документов, которые выпущены рабочей группой 2 технического комитета ИСО/ТК184, данным идентификатором является целая часть номера N.

Элемент **status:** Определяет номенклатуру спецификации на данные с точки зрения ее приемлемости для органа, утверждающего настоящий стандарт.

Примечание 6 — Состояние может принимать следующие значения: 'WD', 'CD', 'DIS', 'FDIS', 'IS', 'TS', 'PAS', 'ITA'.

8.7.2 Спецификация на обмен онтологическими данными повышенного уровня: идентификация протокола обмена представлениями

Протокол обмена представлениями предназначен для определения того, какие внешние библиотечные файлы используются в экземпляре OntoML-документа и какие словарные статьи должны выявляться приемной системой, которая соответствует протоколу обмена представлениями и удовлетворяет дополнительным ограничительным условиям с помощью доставляемого библиотечного файла.

Примечание 1 — Примером протокола обмена представлениями является ИСО 13584-102. Протокол обмена распространяется на представления элементов обмениваемых данных, описанных в библиотеке с помощью представления, совпадающего с одним из прикладных протоколов ИСО 10303.

Идентификатор протокола обмена представлениями определяет подобный частный протокол обмена, представляемый с помощью комплексного XML-типа данных **VIEW_EXCHANGE_PROTOCOL_IDENTIFICATION_Type** (см. рисунок 102).

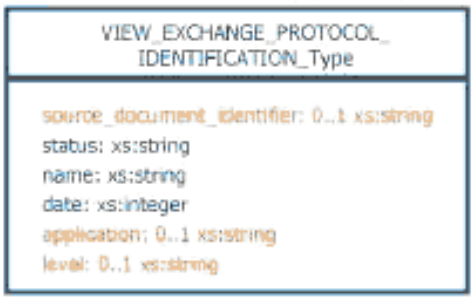


Рисунок 102 — Структура идентификации протокола обмена представлениями

Определения внутренних элементов:

Элемент **application:** Определяет идентификатор для указания характеристик допустимого функционального подмножества полного определения данных в стандартах на протокол обмена представлениями.

Элемент **date:** Определяет год, когда спецификация на дату достигает элемента **status**.

Элемент **level:** Определяет идентификатор, который дополнительно характеризует допустимое подмножество элемента **application**, определенное в стандартах на протокол обмена представлениями.

Элемент **name:** Определяет идентификатор спецификации на данные, в стандартах на протокол обмена представлениями.

Элемент **source_document_identifier:** Определяет идентификатор документа, который содержит спецификацию на данные.

Примечание 2 — Для тех документов, которые выпущены рабочей группой 2 технического комитета ИСО/ТК 184, данным идентификатором является целая часть номера N.

Элемент **status:** Определяет номенклатуру спецификации на данные с точки зрения ее приемлемости для органа, утверждающего настоящий стандарт.

Примечание 3 — Состояние может принимать только следующие значения: 'WD', 'CD', 'DIS', 'FDIS', 'IS', 'TS', 'PAS', 'ITA'.

8.8 Другие структурированные информационные элементы

В данном разделе описываются структурированные информационные элементы, которые используются информационными элементами или понятиями CIIM-онтологии, представленными в предыдущих разделах.

8.8.1 Представление организации

Организация представляет собой административную структуру, представляемую с помощью комплексного XML-типа данных **ORGANIZATION_Type** (см. рисунок 103).

Рисунок 103 — Структура представления организации

Определения внутренних элементов:

Элемент **description:** Определяет текст, который связан с характером организации.

Элемент **id:** Определяет идентификатор для отличия организации.

Элемент **name:** Определяет слово или группу слов, с помощью которых дается ссылка на организации.

8.8.2 Математическая строка

Компонент математической строки предоставляет ресурсы для определения представления математических строк, а также дает представление в MathML-формате. Эта строка представляется с помощью комплексного XML-типа данных **MATHEMATICAL_STRING_Type** (см. рисунок 104).

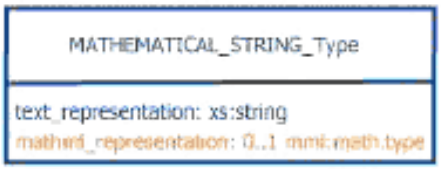


Рисунок 104 — Структура математической строки

Определения внутренних элементов:

Элемент **mathml_representation:** Определяет пометку, выполняемую в соответствии со спецификацией на MathML-язык (версии 2.0, второе издание) (определение типа документа).

Примечание 1 — Спецификацию на MathML-элемент можно получить по адресу: www.w3.org/TR/MathML2.

Элемент **text_representation:** Определяет "линейную" форму математической строки.

Примечание 2 — При необходимости используйте ИСО 843.

Определения внутренних элементов:

Элемент **mml:math.type:** Определяет компонент на MathML-языке, который позволяет определять математические выражения.

Примечание 3 — mml определяет префикс для области имен, связанный со следующей областью MathML-имен: http://www.w3.org/1998/Math/MathML.

8.8.3 Геометрический контекст

Геометрический контекст – это координатное пространство, представляемое с помощью комплексного XML-типа данных **GEOMETRIC_CONTEXT_Type** (см. рисунок 105).

Рисунок 105 — Структура геометрического контекста

Определения внутренних элементов:

Элемент **description:** Определяет эталонные координаты системы объекта для определения элемента **geometric_representation_context**.

Элемент **coordinate_space_dimension:** Определяет положительные целые единицы размерности в координатном пространстве, которые представляет собой геометрический контекст.

8.8.4 Единица измерения геометрического контекста

Единица измерения геометрического контекста определяет те единицы, которые применимы в геометрическом контексте и представляются с помощью комплексного XML-типа данных **GEOMETRIC_UNIT_CONTEXT_Type** (см. рисунок 106).

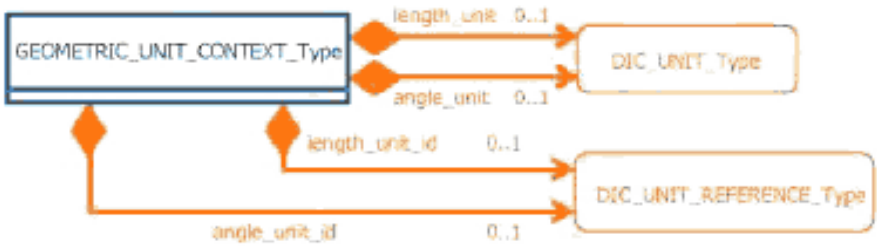


Рисунок 106 — Структура единицы измерения геометрического контекста

Определения внутренних элементов:

Элемент **angle_unit:** Определяет единицу угла, применимую в геометрическом контексте.

Элемент **angle_unit_id:** Определяет идентификатор единицы угла, применимой в геометрическом контексте.

Элемент **length_unit:** Определяет длину единицы, которая применима в геометрическом контексте.

Элемент **length_unit_id:** Определяет идентификатор длины единицы, которая применима в геометрическом контексте.

Примечание 1 — По крайней мере должен быть определен элемент **length_unit**, который может быть связан с элементом **angle_unit**.

Примечание 2 — Если предоставляются оба элемента **angle_unit** и **angle_unit_id**, то элемент **angle_unit** будет обладать приоритетом.

Примечание 3 — Если предоставляются оба элемента **length_unit** и **length_unit_id**, то элемент **length_unit** будет обладать приоритетом.

Определения внешних типов:

Тип **DIC_UNIT_Type:** see 8.4.

Тип **DIC_UNIT_REFERENCE_Type:** См. раздел 8.3.7.

Перечень ограничительных условий:

Предоставляется либо элемент **length_unit,** либо элемент **length_unit_id,** либо оба эти элемента.

9 Структура обмена OntoML-данными

В OntoML-языке определяется множество основанных на онтологии XML-схем обмена онтологическими и каталожными данными. OntoML-язык имеет модульную структуру и содержит множество XML-схем, каждая из которых дает часть модельных CIIM-компонентов.

Помимо логической OntoML-структуры, представленной в разделах 5, 6 и 7, в данном разделе на физическом уровне определены требования, которые должны выполняться с помощью любого экземпляра XML-документа и которые заявляются как представляемые в OntoML-стандарте.

В начале определяется структура идентификатора OntoML CIIM-онтологического понятия, затем – конкретная область имен, позволяющая однозначно определять информационные ресурсы, заимствованные из OntoML-словаря, далее – модульная структура OntoML-языка, и, наконец, - OntoML-подмножества вместе с их классами соответствия.

9.1 Идентификаторы онтологических CIIM-понятий

В настоящем стандарте глобальные идентификаторы используются для идентификации и получения ссылок на онтологические CIIM-понятия. Здесь же определяется их структура и содержание.

OntoML-идентификаторы создаются в соответствии с идентификационной схемой, представленной в ИСО/ТС 29002-5.

Примечание 1 — В ИСО/ТС 29002-5 определяется структура идентификатора, основывающаяся на Международном идентификаторе для регистрации данных (IRDI) согласно ИСО/МЭК 11179-5, в котором глобальный идентификатор, называемый IRDI, присваивается контролируемым элементам, предоставляемым на регистрацию.

Примечание 2 — Все онтологические CIIM-понятия и онтология в целом определяются с помощью IRDI-идентификаторов, которые состоят из трех частей:

— идентификатор регистрирующего органа (RAI), предназначенный для однозначной идентификации организации, ответственной за предоставление идентификаторов данных;

— идентификатор данных (DI), однозначно определяемый для RAI-идентификатора и предназначенный для идентификации контролируемого элемента;

— идентификатор версии (VI), предназначенный для использования с целью управления внесением изменений в контролируемые элементы.

В соответствии с ИСО/ТС 29002-5 входной идентификатор OntoML - онтологии определяется следующей структурой:

ontoml_concept_identifier:= RAI '#' DI '#' VI

В последующих разделах будет рассмотрена структура каждой части IRDI-идентификатора.

9.1.1 Структура идентификатора регистрирующего органа (RAI)

Организация, желающая присвоить идентификаторы онтологическому OntoML-компоненту, должна быть сама по себе однозначно идентифицированной в соответствии с правилами, представленными в ИСО/ТС 29002-5.

Правила в ИСО/ТС 29002-5 определяют структуру для идентификации организаций, состоящую из двух обязательных частей и трех необязательных (дополнительных) частей.

Обязательные части идентификатора организации таковы:

— указатель международного кода (ICD);

— идентификатор организации в рамках идентификационной схемы: элемент данных, содержащий идентификатор организации (OI).

Необязательные (дополнительные) части идентификатора организации таковы:

— идентификатор части организации (OPI): элемент данных, содержащий идентификатор части организации;

— индикатор OPI-источника (OPIS): элемент данных, содержащий значение кода, указывающего на источник OPI;

— дополнительная информация (AI): элемент данных, содержащий информацию, которая присваивается выпускающей организацией.

Если онтология описывается в стандартизированном документе, то требований, указанных в ИСО/ТС 29002-5, становится недостаточно для точного представления идентификатора регистрирующего органа (RAI), поэтому номер стандартного документа должен поставляться как часть общего RAI-идентификатора в части AI.

В подобном случае RAI-идентификатор понятий CIIM-онтологии и онтологий, определенных в стандартизированном документе, будет состоять из следующих трех частей:

— указатель международного кода (ICD), зафиксированный на значении "0112", с помощью которого должна идентифицироваться стандартизирующая организация;

Примечание 1 — ICD-указатель "0112" определен в списке B ИСО 6523 (нумерованном указателе ICD).

— идентификатор стандартизирующей организации (OI) в схеме кодирования "0112".

Примечание 2 — В схеме кодирования (ICD) "0112" код ИСО равен 1, код МЭК равен 2, код ИСО/МЭК равен 3.

— — сам по себе идентификатор стандарта, указанный в части AI, содержит:

— номер стандарта (NB);

— номер детали (PART).

Примечание 3 — Если стандартная деталь не является частью многокомпонентной серии, то номер детали будет представляться с помощью пустой строки.

— номер издания (ED).

Примечание 4 — Идентификация понятий CIIM-онтологии и онтологий, определенных в стандартизированных документах, не требует определения OPI-идентификатора и OPIS-индикатора.

Таким образом, RAI-идентификатор в OntoML-языке определяется в соответствии со следующей синтаксической структурой:

AI ::= NB '_' PART '_' ED

RAI1 ::= ICD '-' OI ('-' OPI ('-' OPIS)?)?

RAI2 ::= ICD '-' OI '- - -' AI

RAI : := RAI 1 | RAI2,

Примечание 5 — Вышеуказанное условие совместимо, но с большими ограничениями, чем в ИСО/ТС 29002-5.

где:

— символ "?" означает дополнительный оператор;

— символ "|" означает логический оператор дизъюнкции;

— символы скобок являются символами мета-обозначений для определения групп;

— символы между кавычками (') – это символы ограничения, которые могут появляться в RAI-идентификаторе;

— идентификаторы вне кавычек, которые должны иметь следующую структуру:

— символ ICD: строка, содержащая ровно четыре цифры.

Примечание 6 — Предшествующая цифра 0 присоединяется к ICD-указателю в том случае, когда он содержит менее 4 символов.

— OI-идентификатор: строка, содержащая до 35 буквенно-цифровых символов;

— OPI-идентификатор: строка, содержащая до 35 буквенно-цифровых символов;

— OPIS-индикатор: единственный цифровой символ;

— NB-номер: строка, содержащая до 10 буквенно-цифровых символов.

Примечание 7 — В стандартах ИСО и МЭК NB-номер будет содержать только цифры;

— PART-номер: строка (которая может быть пустой), содержащая до 10 буквенно-цифровых символов.

Примечание 8 — В стандартах ИСО и МЭК PART-номер будет содержать только цифры.

— ED-номер: строка, содержащая до пяти цифр.

Примечание 9 — В стандартах ИСО и МЭК ED-номер будет содержать только цифры.

Пример 1 — Французская компания "FOO" в соответствии с французской системой кодировки компаний (ICD="0002") идентифицируется (ORGID) с помощью идентификатора-номера SIRENE "12345678901234". Соответствующий идентификатор организации теперь будет иметь следующий вид:

0002-12345678901234

Пример 2 — Идентификатором (ORGID), относящимся к крупным производителям устройств, является "123456789", который присваивается как DUNS-идентификационный номер компаний (ICD="0060"). Конкретный завод производителя, находящийся в «Городе» также идентифицируется OPI с помощью следующего идентификатора: "12345". Этот OPI-идентификатор присваивается самой компанией, причем OPIS-индикатору присваивается значение 1. Соответствующий идентификатор организации будет иметь следующий вид:

0060-123456789-12345-1

Пример 3 — МЭК 61360-4 – это редактируемый стандарт МЭК, в котором определяется словарь-справочник данных относительно электронных компонентов. МЭК идентифицирует (ORGID="2") в соответствии с ИСО-регистром стандартов, разрабатываемых организациями (ICD="112"). Данный номер стандарта (NB) равен "61360", а конкретная деталь (PART), для которой определен словарь данных, имеет номер "4". Если мы предположим, что необходима ссылка на первую редакцию данного документа (ED="1"), то соответствующий идентификатор примет следующий вид:

0112-2---61360_4_1

9.1.2 Структура идентификатора редакции (VI)

Каждое понятие CIIM-онтологии претерпевает несколько редакций в течение всего его жизненного цикла. Для однозначной идентификации редакции каждого понятия необходимо также представлять эту редакцию.

Идентификатор версии (VI) в OntoML-языке определяется следующим образом:

— VI-идентификатор: строка, содержащая до 10 цифр.

Примечание — Вышеуказанное условие совместимо, но имеет большие ограничения, чем ИСО/ТС 29002-5.

9.1.3 Структура идентификатора данных (DI)

Каждый онтологический компонент, определенный в данной онтологии, должен снабжаться идентификатором, называемым «идентификатором данных» и определяемым идентифицированным органом регистрации, поэтому предполагается, что каждый идентификатор данных, присвоенный понятию экземпляра класса в CIIM-онтологии, должен быть уникальным в контексте данного органа регистрации.

В OntoML-языке идентификатор данных (DI) определяется следующим образом:

— идентификатор кодового пространства (CSI): идентификатор области, внутри которой каждый код будет означать одно и то же содержание;

Пример — В OntoML-языке класс и свойство могут иметь один и тот же код. С помощью двух различных идентификаторов кодового пространства для кодов класса и свойства можно сделать полученные идентификаторы однозначными.

Примечание 1 — В ИСО/ТС 29002-5 определен список идентификаторов зарегистрированного кодового пространства.

— код элемента (IC);

OntoML-идентификатор данных определяется в соответствии со следующей синтаксической структурой:

DI ::= CSI '-' IC,

где:

— CSI: строка, содержащая только два буквенно-цифровых символа:
  - "01": идентификатор кодового пространства для класса;
  - "02": идентификатор кодового пространства для свойства;
  - "04": идентификатор кодового пространства для ограничительного условия;
  - "05": идентификатор кодового пространства для единицы измерений;
  - "07": идентификатор кодового пространства для кода значений;
  - "08": идентификатор кодового пространства для валюты;
  - "09": идентификатор кодового пространства для типа данных;
  - "10": идентификатор кодового пространства для документа;
  - "11": идентификатор кодового пространства для онтологии или библиотеки;

Примечание 2 — CSI-идентификатор определен в соответствии с ИСО/ТС 29002-5.

— IC-код: строка, содержащая до 71 буквенно-цифровых символа.

9.1.3.1 Классы идентификатора данных (DI)

В соответствии с ИСО/ТС 29002-5 класс связывается с идентификатором кодового пространства (CSI), равным '01'. DI-идентификатор для класса состоит из следующих трех частей:

— идентификатора кодового пространства для данного класса;
— символа дефиса ("-");
— кода данного класса.

Пример — В ИСО 13584-511 определена онтология, связанная с крепежными элементами. В этой онтологии определен класс характеристик продукции, называемый "болт с шестигранной головкой". CSI-идентификатор для этого класса равен "01", IC-код - "P511AAA156", а идентификатор редакции (VI) устанавливается на "1". При этом соответствующий идентификатор класса характеристик продукции будет иметь следующий вид:

0112-1---13584_511_1#01-P511AAA156#1

9.1.3.2 Идентификаторы данных (DI) для свойств, типов данных и документов

Идентификаторы данных (DI) для свойств, типов данных и документов определяются с помощью:

— идентификатора кодового пространства для свойства, типа данных или документа;
— символа дефиса ("-");
— кода свойства, типа данных и документа.

Пример — В ИСО 13584-501 определена онтология, связанная с измерительными инструментами. В классе характеристик продукции "лабораторный измерительный инструмент" определяется свойство, называемое "классом точности". В соответствии с ИСО/ТС 29002-5 идентификатор кодового пространства для свойства (CSI) равен "02". Это

свойство идентифицируется с помощью IC-кода "P501_P000178" и связывается с VI-кодом редакции, равным "000000001". Соответствующий идентификатор свойства будет иметь следующий вид:

112-1---13584_501_1#02-P501_P0001 78#000000001

9.1.3.3 Идентификаторы данных (DI) для единиц измерений, валют, ограничительных условий, кодов значений и *апостериорных* сематических отношений

OntoML-язык позволяет присваивать IRDI-идентификаторы единицам измерений, валютам, ограничительным условиям, кодам значений и *апостериорным* семантическим соотношениям. Единицам измерений, валютам, ограничительным условиям и кодам значений присваивается RAI-идентификатор.

Идентификаторы данных для единиц измерений, валют, ограничительных условий, кода значений или апостериорного семантического соотношения состоят из следующих трех частей:

— идентификатора кодового пространства для единиц измерений, валют, ограничительных условий, кода значений или апостериорного семантического соотношения;

— символа дефиса ("-");

— кода единицы измерений, валюты, ограничительного условия, кода значений или апостериорного семантического соотношения.

9.1.3.4 Идентификаторы данных (DI) для онтологий и библиотек

OntoML-язык также позволяет присваивать IRDI-идентификаторы онтологии или библиотеке. Им также может присваиваться и RAI-идентификатор. Единственное ограничительное условие состоит в том, что этот код является уникальным для всех классов, онтологий и библиотек, определенных с помощью данного RAI-идентификатора. Этот код состоит из идентификатора данных (DI) для онтологии или библиотеки.

Идентификатор данных (DI) для онтологии или библиотеки состоит из следующих трех частей:

— идентификатора кодового пространства для онтологии;

— символа дефиса ("-");

— кода онтологии или библиотеки.

Пример — Первая редакция онтологии, идентифицированная с помощью кода "99999", определяется крупным производителем устройств. В соответствии с ИСО/ТС 29002-5 идентификатор кодового пространства (CSI) равен "11". Указанный производитель идентифицируется как код "123456789", который распространяется как DUNS-номер для идентифицированных компаний (ICD="0060"). Также идентифицируется и конкретный завод производителя, расположенный в "Городе" с помощью следующего OPI-идентификатора: "12345". Поскольку этот OPI-идентификатор распространяется самой компанией-производителем, то OPIS-индикатор устанавливается на 1. Соответствующий онтологический идентификатор при этом выражается следующим образом:

0060-123456789-12345-1 #11 -99999#1

9.1.4 Представление OntoML-идентификатора

Идентификаторы онтологических OntoML CIIM-понятий представляются с использованием обязательного XML-атрибута, присваиваемого соответствующему понятию CIIM-онтологии. Имя этого атрибура – всегда **id.** Его конкретный тип данных зависит от вида понятия CIIM-онтологии, которое предназначено для идентификации. Этими типами являются:

— Тип **SupplierId** – для онтологического понятия поставщика;

— Тип **ClassId** – для онтологического понятия класса;

— Тип **PropertyId** – для онтологического понятия свойства;

— Тип **DatatypeId** – для онтологического понятия типа данных;

— Тип **DocumentId** – для онтологического понятия документа.

Кроме того, тип идентификатора онтологии и/или библиотеки, единицы измерений, ограничительного условия, кода значений или *апостериорного* семантическогро соотношения определяется следующими типами:

— Тип **OntologyId** – для онтологии или библиотеки;

— Тип **DicUnitId** – для единицы измерений;

— Тип **CurrencyId** – для валюты;

— Тип **ConstraintId** – для ограничительного условия;

— Тип **ValueCodeId** – для кода значения;

— Тип **APosterioriSemanticRelationId** – для *апостериорного* семантического соотношения.

9.2 Область OntoML-имен

В OntoML-языке определяется область имен, основанная на URN- и URI-спецификациях.

OntoML-процессоры должны использовать методологию XML-областей имен для выявления элементов и атрибутов этих областей. Поставщики не должны расширять OntoML-области имен дополнительными элементами или атрибутами.

Данное описание не использует каких-либо префиксов для ссылки на элементы OntoML-области имен, однако OntoML-документы свободны от использования каких-либо префиксов при условии, что имеется определение области имен, которое связывает префикс с унифицированным идентификатором ресурса (URI) для OntoML-области имен.

9.2.1 Унифицированное имя OntoML-ресурса (URN)

OntoML-область имен обладает следующим унифицированным именем ресурса (URN):
urn:iso:std:iso:13584:-32:ed-1:tech:xml-schema:ontoml

Примечание — Унифицированное имя ресурса (URN) определено в соответствии с ИСО URN-схемой для ресурсов, определенных в стандартах ИСО/ТК184/ПК4.

9.2.2 Унифицированный идентификатор OntoML-ресурса (URI)

Область имен в OntoML-языке имеет следующий унифицированный идентификатор ресурса (URI):
http://www.tc1 84-sc4.org/2010/OntoML

Примечание — Часть строки "2010" в URI-идентификаторе указывает год, в котором этот идентификатор был размещен, а не вариант используемого OntoML-модуля, который определяется с помощью атрибутов.

9.3 Модульная структура OntoML-языка

OntoML-язык является набором из XML-схем, называемых «модулями», которые определяют общие ресурсы, на которых создается схема высокого уровня. Ниже приведен список этих модулей:

— Модуль **ontoml.xsd:** Является основной XML-схемой, определяющей верхний уровень в формате обмена данными OntoML-онтологии и основанной на определении одиночного XML-элемента **ontoml**, чье содержание является обязательным заголовком, за которым следует дополнительное описание словаря и допонительное содержание;

— Модуль **header.xsd:** Содержит управляющие ресурсы для указания характеристик содержания (дат, авторов и т.п.) в OntoML XML-файле;

— Модуль **dictionary.xsd:** Определяет хранилище для любой CIIM-совместимой онтологии;

— Модуль **supplier.xsd:** Содержит ресурсы, предназначенные для представления источников информации в соответствии со структурой, определенной в разделе 6.7.1;

— Модуль **class.xsd:** Содержит ресурсы, предназначенные для представления класса онтологии в соответствии со структурой, определенной в разделе 6.7.2, независимо от ее характера (класс общих моделей, класс функциональных моделей или класс функциональных представлений);

— Модуль **property.xsd:** Содержит ресурсы, предназначенные для представления свойства, определенного в разделе 6.7.4, независимо от его характера (характеристическое свойство, контекстное свойство, контекстно-зависимое свойство или свойство экземпляра);

— Модуль **datatype.xsd:** Содержит ресурсы, предназначенные для представления типов данных, определенных в разделе 6.7.6;

— Модуль **document.xsd:** Содержит ресурсы, предназначенные для представления документа, определенного в разделе 6.7.7;

— Модуль **type_system.xsd:** Содержит XML-представление полной OntoML-системы типов, представленной в разделе 8.3;

— Модуль **translation.xsd:** Содержит ресурсы, предназначенные для определения многоязычных представлений незашифрованного текста, указанного в разделе 8.1;

— Модуль **external_file.xsd:** Содержит структурные элементы для ссылок на внешние ресурсы, которые могут либо обмениваться вместе с экземплярами OntoML-документа, либо ссылаться на Интернет; эти элементы указаны в разделе 8.2;

— Модуль **unit.xsd:** Содержит структурные элементы для точного описания любого вида единиц в соответствии со структурой, определенной в разделе 8.4;

— Модуль **a_posteriori.xsd:** Содержит структурные элементы для представления *апостериорного* установления соответствия между классами и свойствами, определенными в различных онтологиях в соответствии со структурой, определенной в разделе 8.6;

— Модуль **constraint.xsd:** Содержит структурные элементы для представления ограниченных свойств в соответствии со структурой, определенной в разделе 8.5;

— Модуль **basic.xsd:** Содержит все простые и комплексные определения OntoML-типов данных, совместно используемых OntoML-модулями;

— Модуль **identifier.xsd:** Предоставляет структурные элементы, необходимые для

идентификации и ссылок на понятия CIIM-онтологии, представленные в разделе 9.1;

— Модуль **content.xsd:** Определяет структуру семейства продукции, принадлежащей данной библиотеке;

— Модуль **library.xsd:** Определяет структуру для представления библиотек с семействами продукции с помощью ее характеристик, возможно, связанной с онтологией, в которой определены классы характеристик и свойств.

Таблица 1 с помощью взаимных ссылок иллюстрирует связь между рассмотренными выше модулями, причем выделенные цветом клетки указывают на взаимосвязь одного модуля с другим.

Таблица 1 – Взаимные ссылки между OntoML-модулями

references →	basic.xsd	identifier.xsd	translation.xsd	unit.xsd	external_file.xsd	header.xsd	supplier.xsd	type_system.xsd	datatype.xsd	constraint.xsd	property.xsd	class.xsd	document.xsd	a_posteriori.xsd	dictionary.xsd	content.xsd	library.xsd	ontoml.xsd
basic.xsd	■																	
identifier.xsd		■																
translation.xsd	■	■	■															
unit.xsd	■			■														
external_file.xsd	■	■	■	■	■													
header.xsd		■	■		■	■												
supplier.xsd	■	■	■				■											
type_system.xsd	■	■	■	■	■			■										
datatype.xsd	■	■	■		■			■	■									
constraint.xsd	■	■						■		■								
property.xsd	■	■	■		■			■			■							
class.xsd	■	■	■		■					■		■						
document.xsd	■	■	■		■								■					
aPosteriori.xsd		■												■				
dictionary.xsd	■	■					■		■		■	■	■	■	■			
content.xsd	■	■		■	■											■		
library.xsd		■														■	■	
ontoml.xsd						■									■		■	■

References – Ссылки.

9.4 Уровни обмена данными и классы соответствия

OntoML-язык объединяет в себе множество структурированных ресурсов в единую XML-схему для представления онтологий и, возможно, библиотек поставщика с целью обмена данными. Для поддержки требований к обмену данными на различных уровнях необходимо идентифицировать в OntoML-языке четыре функциональных подмножества – допустимые уровни обмена данными. Эти различные функциональные подмножества называются «классами соответствия».

Эти классы соответствия также определяют допустимые уровни OntoML-реализации для тех систем, которые заявляются как соответствующие стандарту на OntoML-язык.

Четыре класса соответствия в OntoML-языке определяются следующим образом:

— простая (элементарная) онтология: онтология, которая определяет иерархии классов элементов на основе общей ИСО/МЭК-схемы словаря вместе с требуемыми внешними ресурсами, документами, типами данных и группами типов данных, но без описания представлений элементов (т.е. классов функциональных моделей) и категорий представления элементов (т.е. классов функциональных представлений), соответствующих OntoML-классу соответствия 1;

— усложненная онтология: онтология, которая соответствует простой онтологии, но с дополнением в виде ресурсов для определения иерархий представлений элементов (т.е. классов функциональных моделей) и категорий представления элементов (т.е. классов функциональных экземпляров), соответствующих OntoML-классу соответствия 1;

— простая (элементарная) библиотека: содержание библиотеки, определяющее продукцию на основе простых онтологий, возможно, обмениваемых вместе с определениями простой онтологии, соответствующих OntoML-классу соответствия 3;

— усложненная библиотека: содержание библиотеки, определяющее продукцию и ее представления на основе усложненных онтологий, возможно, обмениваемых вместе с определениями усложненной онтологии, соответствующих OntoML-классу соответствия 4.

В таблице 2 приведена сводка поддерживаемых возможностей различных классов соответствия OntoML-языка.

Таблица 2 – Дополнительные варианты соответствия между классами соответствия и опциями соответствия OntoML-языка

	Онтология		Библиотека	
Класс соответствия	Общие определения словаря согласно ИСО/МЭК Документы группы внешних ресурсов	Представления категории представлений	Библиотека продукции	Библиотека представлений
1	X			
2	X	X		
3	X		X	
4	X	X	X	X

В приложении С определены стандартные данные, которые позволяют идентифицировать один класс соответствия среди всех других подобных классов.

9.5 Требование к классам соответствия

9.5.1 Класс соответствия 1

В классе соответствия 1 рассматриваются те реализации, которые поддерживают онтологии, определяющие иерархии классов элементов на основе общей ИСО/МЭК-схемы словаря, вместе с требуемыми внешними ресурсами, онтологическими понятиями типов данных и групп типов данных. Словарь должен поддерживать стандартизированные данные, определенные в приложении С, а также следующие комплексные XML-типы данных:

— В модуле ***ontoml.xsd***:
  — Тип **ONTOML_Type**

— В модуле ***identifiers.xsd***:
  — Тип **CLASS_REFERENCE_Type**
  — Тип **CLASSES_REFERENCE_Type**
  — Тип **DATATYPE_REFERENCE_Type**
  — Тип **DATATYPES_REFERENCE_Type**

- Тип **DIC_UNIT_REFERENCE_Type**
- Тип **DIC_UNITS_REFERENCE_Type**
- Тип **DICTIONARY_REFERENCE_Type**
- Тип **DICTIONARIES_REFERENCE_Type**
- Тип **DOCUMENT_REFERENCE_Type**
- Тип **DOCUMENTS_REFERENCE_Type**
- Тип **PROPERTY_REFERENCE_Type**
- Тип **PROPERTIES_REFERENCE_Type**
- Тип **SUPPLIER_REFERENCE_Type**
- Тип **SUPPLIERS_REFERENCE_Type**

Примечание 1 — **idt:IRDI**-идентификатор определен вне OntoML-языка. Префикс ***idt*** обозначает область имен, идентифицированную с помощью следующего унифицированного имени ресурса (URN): *urn:iso:std:iso:29002:-5:ed-1:tech:schema:identifier*.

- В модуле ***header.xsd***:
  - Тип **HEADER_Type**
  - Тип **INFORMATION_Type**
  - Тип **LIBRARY_IIM_IDENTIFICATION_Type**
- В модуле ***dictionary.xsd***:
  - Тип **A_POSTERIORI_SEMANTIC_RELATIONSHIPS_Type**
  - Тип **CONTAINED_CLASSES_Type**
  - Тип **CONTAINED_DOCUMENTS_Type**
  - Тип **CONTAINED_DATATYPES_Type**
  - Тип **CONTAINED_PROPERTIES_Type**
  - Тип **CONTAINED_SUPPLIERS_Type**
  - Тип **DICTIONARY_IN_STANDARD_FORMAT_Type**
  - Тип **DICTIONARY_Type**
- В модуле ***supplier.xsd***:
  - Тип **SUPPLIER_Type**
- В модуле ***class. xsd***:
  - Тип **ASSIGNED_VALUE_Type**
  - Тип **CLASS_CONSTANT_VALUES_Type**
  - Тип **CLASS_Type**
  - Тип **CLASS_VALUE_ASSIGNMENT_Type**
  - Тип **CATEGORIZATION_CLASS_Type**
  - Тип **ITEM_CLASS_CASE_OF_Type**
  - Тип **ITEM_CLASS_Type**
  - Тип **val:value**

Примечание 2 — Тип **val:value** определен вне OntoML-языка. Префикс *cat* обозначает область имен, идентифицированную с помощью следующего унифицированного имени ресурса (URN): *urn:iso:std:iso:29002:-10:ed-1:tech:schema:catalogue*.

- В модуле ***property.xsd***:
  - Тип **CONDITION_DET_Type**
  - Тип **DEPENDENT_P_DET_Type**
  - Тип **NON_DEPENDENT_P_DET_Type**

- Тип **PROPERTY_Type**
- Тип **SYNONYMOUS_SYMBOLS_Type**

— В модуле ***datatype.xsd***:
- Тип **DATATYPE_Type**

— В модуле ***datatypeSystem.xsd***:
- Тип **ALTERNATIVE_UNITS_Type**
- Тип **ANY_TYPE_Type**
- Тип **ARRAY_TYPE_Type**
- Тип **BAG_TYPE_Type**
- Тип **BOOLEAN_TYPE_Type**
- Тип **CLASS_REFERENCE_TYPE_Type**
- Тип **DATE_DATA_TYPE_Type**
- Тип **DATE_TIME_DATA_TYPE_Type**
- Тип **DIC_VALUE_Type**
- Тип **INT_CURRENCY_TYPE_Type**
- Тип **INT_MEASURE_TYPE_Type**
- Тип **INT_TYPE_Type**
- Тип **INT_DIC_VALUE_Type**
- Тип **ITS_VALUES_Type**
- Тип **LEVEL_Type**
- Тип **LEVEL_TYPE_Type**
- Тип **LIST_TYPE_Type**
- Тип **NAMED_TYPE_Type**
- Тип **NON_QUANTITATIVE_CODE_TYPE_Type**
- Тип **NON_QUANTITATIVE_INT_TYPE_Type**
- Тип **NON_TRANSLATABLE_STRING_TYPE_Type**
- Тип **NUMBER_TYPE_Type**
- Тип **REAL_CURRENCY_TYPE_Type**
- Тип **REAL_MEASURE_TYPE_Type**
- Тип **REAL_TYPE_Type**
- Тип **REMOTE_HTTP_ADDRESS_Type**
- Тип **REPRESENTATION_REFERENCE_TYPE_Type**
- Тип **SET_TYPE_Type**
- Тип **SET_WITH_SUBSET_CONSTRAINT_TYPE_Type**
- Тип **STRING_DIC_VALUE_Type**
- Тип **STRING_TYPE_Type**
- Тип **TIME_DATA_TYPE_Type**
- Тип **TRANSLATABLE_STRING_TYPE_Type**

— В модуле ***translations.xsd***:
- Тип **DOCUMENT_IDENTIFIER_NAME_LABEL_Type**
- Тип **DOCUMENT_IDENTIFIER_Type**

- Тип **GENERAL_TEXT_Type**
- Тип **KEYWORD_LABEL_Type**
- Тип **KEYWORD_Type**
- Тип **LANGUAGE_Type**
- Тип **PREFERRED_NAME_LABEL_Type**
- Тип **PREFERRED_NAME_Type**
- Тип **SHORT_NAME_LABEL_Type**
- Тип **SHORT_NAME_Type**
- Тип **SYNONYMOUS_NAME_LABEL_Type**
- Тип **SYNONYMOUS_NAME_TYPE_Type**
- Тип **TEXT_Type**
- Тип **TRANSLATION_DATA_Type**
- Тип **TRANSLATION_Type**

— В модуле ***externalFiles.xsd***:
- Тип **EXTERNAL_GRAPHICS_Type**
- Тип **EXTERNAL_RESOURCE_Type**
- Тип **EXTERNAL_FILES_Type**
- Тип **GRAPHICS_Type**
- Тип **HTTP_FILE_Type**
- Тип **IDENTIFIED_DOCUMENT_Type**
- Тип **REFERENCED_DOCUMENT_Type**
- Тип **REFERENCED_GRAPHICS_Type**
- Тип **SOURCE_DOCUMENT_Type**

— В модуле ***units.xsd***:
- Тип **CONTEXT_DEPENDENT_UNIT_Type**
- Тип **CONVERSION_BASED_UNIT_Type**
- Тип **DERIVED_UNIT_ELEMENT_Type**
- Тип **DERIVED_UNIT_Type**
- Тип **DIC_UNIT_Type**
- Тип **DIMENSIONAL_EXPONENTS_Type**
- Тип **NAMED_UNIT_Type**
- Тип **NON_SI_UNIT_Type**
- Тип **SI_UNIT_Type**
- Тип **UNIT_Type**

— В модуле ***constraints.xsd***:
- Тип **CARDINALITY_CONSTRAINT_Type**
- Тип **CLASS_CONSTRAINT_Type**
- Тип **CONFIGURATION_CONTROL_CONSTRAINT_Type**
- Тип **CONSTRAINT_OR_CONSTRAINT_ID_Type**
- Тип **CONSTRAINT_Type**
- Тип **CONSTRAINTS_Type**

- Тип **CONTEXT_PARAMETER_CONSTRAINTS_Type**
- Тип **CONTEXT_RESTRICTION_CONSTRAINT_Type**
- Тип **DOMAIN_CONSTRAINT_Type**
- Тип **ENUMERATION_CONSTRAINT_Type**
- Тип **FILTER_Type**
- Тип **INTEGRITY_CONSTRAINT_Type**
- Тип **POSTCONDITION_Type**
- Тип **PRECONDITION_Type**
- Тип **PROPERTY_CONSTRAINT_Type**
- Тип **RANGE_CONSTRAINT_Type**
- Тип **STRING_PATTERN_CONSTRAINT_Type**
- Тип **STRING_SIZE_CONSTRAINT_Type**
- Тип **SUBCLASS_CONSTRAINT_Type**
- Тип **SUBSET_Type**

— В модуле ***baseTypes.xsd***:
- Тип **MATHEMATICAL_STRING_Type**
- Тип **ORGAN IZATION_Type**
- Тип **mml:math.type**

Примечание 3 — Тип **mml:math.type** определяется вне OntoML-языка. Префикс *mml* обозначает область имен, идентифицированную с помощью следующего унифицированного имени ресурса (URN) http://www.w3.org/1998/Math/MathML:

— В модуле ***document.xsd***:
- Тип **AUTHORS_Type**
- Тип **DOCUMENT_CONTENT_Type**
- Тип **DOCUMENT_Type**
- Тип **PERSON_Type**
- Тип **REMOTE_LOCATIONS_Type**
- Тип **STRINGS_Type**

— В модуле ***aPosteriori.xsd***:
- Тип **A_POSTERIORI_CASE_OF_Type**
- Тип **A_POSTERIORI_SEMANTIC_RELATIONSHIP_Type**
- Тип **CORRESPONDING_PROPERTIES_type**
- Тип **MAPPING_FUNCTION_Type**
- Тип **PROPERTY_MAPPING_Type**

9.5.2 Класс соответствия 2

В классе соответствия 2 рассматриваются те реализации, которые поддерживают онтологии, определяющие иерархии классов элементов на основе общей ИСО/МЭК-схемы словаря, вместе с требуемыми внешними ресурсами, онтологическими понятиями документа типов данных, описанием иерархии представлений элементов и их категорий. Он должен поддерживать стандартизированные данные, определенные в приложении С, а также комплексные типы данных и связанные с ними структурные элементы, определенные в классе соответствия 1 и, кроме того, следующие комплексные типы данных и связанные с ними структурные элементы:

— В модуле ***header.xsd***:
- Тип **SUPPORTED_VEP_Type**

  - Тип **VIEW_EXCHANGE_PROTOCOL_IDENTIFICATION_Type**
- В модуле ***class.xsd***:
  - Тип **FM_CLASS_VIEW_OF_Type**
  - Тип **FUNCTIONAL_MODEL_CLASS_Type**
  - Тип **GEOMETRIC_CONTEXT_Type**
  - Тип **GEOMETRIC_UNIT_CONTEXT_Type**
  - Тип **NON_INSTANTIABLE_FUNCTIONAL_VIEW_CLASS_Type**
  - Тип **V_C_V_RANGE_Type**
  - Тип **VIEW_CONTROL_VARIABLE_RANGE_Type**
- В модуле ***property.xsd***:
  - Тип **REPRESENTATION_P_DET_Type**
- В модуле ***datatypeSystem.xsd***:
  - Тип **AXIS1_PLACEMENT_TYPE_Type**
  - Тип **AXIS2_PLACEMENT_3D_TYPE_Type**
  - Тип **AXIS2_PLACEMENT_2D_TYPE_Type**
  - Тип **PLACEMENT_TYPE_Type**
  - Тип **PROGRAM_REFERENCE_TYPE_Type**
  - Тип **REPRESENTATION_REFERENCE_TYPE_Type**
- В модуле ***aPosteriori.xsd***:
  - **A_POSTERIORI_VIEW_OF_Type**

9.5.3 Класс соответствия 3

В классе соответствия 3 рассматриваются те реализации, которые поддерживают обмен данными о продукции вместе со словарными определениями. Он должен поддерживать стандартизированные данные, определенные в приложении С, а также комплексные типы данных и связанные с ними структурные элементы, определенные в классе соответствия 1 и, кроме того, следующие комплексные типы данных и связанные с ними структурные элементы:

- В модуле ***externalFiles.xsd***:
  - Тип **ILLUSTRATION_Type**
  - Тип **MESSAGE_Type**
- В модуле ***library.xsd***:
  - Тип **CONTAINED_CLASS_EXTENSIONS_Type**
  - Тип **LIBRARY_IN_STANDARD_FORMAT_Type**
  - Тип **LIBRARY_Type**
- В модуле ***content.xsd***:
  - Тип **CLASS_EXTENSION_Type**
  - Тип **CLASS_PRESENTATION_ON_PAPER_Type**
  - Тип **CLASS_PRESENTATION_ON_SCREEN_Type**
  - Тип **CLASSIFICATION_Type**
  - Тип **CREATE_ICON_Type**
  - Тип **EXPLICIT_ITEM_CLASS_EXTENSION_Type**
  - Тип **PROPERTY_VALUE_RECOMMENDED_PRESENTATION_Type**
  - Тип **PROPERTY_CLASSIFICATION_Type**
  - Тип **RECOMMENDED_PRESENTATION_Type**
  - Тип **cat:catalogue_Type**

Примечание — Тип **cat:catalogue_Type** определяется вне OntoML-языка. Префикс *cat* обозначает область имен, идентифицированную с помощью следующего унифицированного имени ресурса (URN): *urn:iso:std:iso:29002:-10:ed-1*.

9.5.4 Класс соответствия 4

В классе соответствия 4 рассматриваются те реализации, которые поддерживают обмен данными о продукции и инженерными моделями вместе со словарными определениями. Он должен поддерживать стандартизированные данные, определенные в приложении С, а также комплексные типы данных и связанные с ними структурные элементы, определенные в классе соответствия 3.

— В модуле ***content.xsd***:
  — Тип **CONTEXT_PARAM_ICON_Type**
  — Тип **EXPLICIT_FUNCTIONAL_MODEL_CLASS_EXTENSION_Type**

10 Правила управления внесением изменений в словарь

В данном разделе определены правила организации, контроля и отслеживания изменений в словарях-справочниках.

Задав конкретную версию O_t словаря-справочника, а также множество описаний продукции, основанное на версии O_r этого словаря, отметим, что назначением этих правил является (1) принятие решения о том, является ли словарь O_t доступным для правильной интерпретации продукции, основанной на O_r, (2) в противном случае – для принятия решения о том, какая часть словаря O_t должна быть обновлена для правильной интерпретации доступных описаний продукции.

Примечание — Правила управления словарем, представленные в данном разделе, основываются на характеристиках продукции. Классы, не используемые для определения характеристик продукции, не должны рассматриваться в дискуссии, поэтому в данном разделе термин «класс» будет означать «класс характеристик», а термин «онтологические понятия» - «все онтологические понятия, но принадлежащие классам характеристик». Правило для класса характеристик определено в правиле 8.

10.1 Принцип онтологической непрерывности

Роль онтологии в конктретной области комплекса международных стандартов ИСО 13584, называемой «словарем-справочником» для этой области, состоит в обеспечении:

— обмена однозначной информации относительно продукции между бизнес-партнерами, и
— сохранения неизменных характеристик продукции в различных постоянно пополняемых хранилищах.

Используемый для этого метод состоит в кодировании каждой продукции с помощью характеристик и содержит:

— класс характеристик, к которому принадлежит данная продукция, и
— множество пар «свойство/значение», в котором свойства выбираются из тех свойств, которые применимы к данному классу характеристик.

Пример — Если продукция представляется с использованием словаря-справочника для крепежных элементов (согласно ИСО 13584-511), то для колпачковой гайки с номинальным диаметром 5 и высотой 4 имеем следующую сокращенную запись:

колпачковая гайка (номинальный диаметр = 5; высота гайки = 4),

Эти характеристики представляют продукцию как "шестигранную гайку, заканчивающуюся с одной стороны плоской крышкой", чей "номинальный диаметр резьбовой части" составляет 5 мм, а "общая высота гайки" - 4 мм.

Примечание 1 — В формате обмена данными между компьютерами, класс характеристик и свойства, появляющиеся в характеристиках, представляются с использованием кодов, которые включают в себя номер версии этих онтологических понятий. В вышеприведенном примере определение характеристик представляется с помощью предпочтительных имен классов и свойств с целью достижения их более полного понимания.

Примечание 2 — В основанном на онтологии подходе каждый вид продукции представляется как экземпляр онтологии.

Фундаментальное предположение, на котором основывается кодирование, таково:

— при обмене данными и отправитель, и получатель данных должны связывать одно и то же содержание с одними и теми же характеристиками, и
— характеристики, зарегистрированные в момент времени = t, должны интерпретироваться с одним и тем же содержанием в момент времени = t+1, даже если словарь изменится в промежутке времени от t до t+1.

В общем случае существует два решения, которые позволяют соблюсти это фундаментальное предположение, а именно:

— В случае, когда словарь претерпевает изменения между моментами времени t и t+1, и когда

ограничения на внесение допустимых изменений отсутствуют, пользователь словаря должен быть способен получать доступ к словарю как в момент времени t, так и в момент времени t+1, т.е. к различным его версиям;

— Другое решение состоит в сохранении словаря (с помощью правил управления внесением изменений в словарь, определенных в настоящем стандарте), что позволит использовать лишь одну версию всех онтологических понятий.

Последнее из решений, называемое «*принципом онтологической неразрывности*», позволяет ограничивать допустимые изменения до тех, которые будут гарантировать, что любые характеристики продукции, определенные в момент времени = t в существующем в это время словаре, будут сохраняться с неизменным содержанием при их интерпретации с помощью словаря, который будет существовать в момент времени = t+1. Следовательно, содержание (смысл) онтологического понятия, введенное в какой-либо момент времени, будет сохраняться и в будущем.

Примечание 3 — В течение всего времени жизни описания в словаре могут содержать небольшие ошибки типа опечаток, которые также потребуется устранить, например, для учета технологических усовершенствований. Наконец, может также наступить момент, когда в определениях словаря появятся концептуальные ошибки, при которых содержание классов и/или свойств потребует изменений.

Правила внесения изменений в словаре, рассмотренные в данном разделе, позволяют классифицировать различные изменения, возникающие в течение всего времени жизни словарей-справочников. Здесь также определяется, каким образом каждое изменение должно проводиться для гарантии того, что одно и то же содержание словаря всегда будет связано с одной и той же существующей характеристикой.

10.2 Редакции и версии словаря

Последствия от внесения изменений зависят от их влияния на существующие и последующие характеристики. Первоначально необходимо определить, что означает факт соответствия характеристикам словаря.

Пусть:

— O_t – словарь O в момент времени t;

— C_t – классы словаря O в момент времени t;

— P_t – свойства в словаре O в момент времени t;

— applicable_properties$_t$ – функция, связанная с каждым классом C_t и применимыми свойствами P_t в момент времени t.

Примечание 1 — Функция **applicable_properties$_t$** (**Class_Pt**) представляет все свойства, заявляемые с помощью XML-элемента **described_by** комплексного XML-типа данных **CLASS_Type,** который определяет класс **Class_P$_t$**, если класс **Class_P$_t$** является условным классом. Свойства, импортированные с помощью элемента **imported_properties** класса **Class_P$_t$**, являются применимыми свойствами возможных суперклассов класса **Class_P$_t$**.

— **domain$_t$** – функцией, которая связана с каждым свойством Pi его области значений P_t в момент времени t.

Примечание 2 — Функция **domain$_t$**(Pi_t) представляет собой область значений, заявляемую как XML-элемент области значений комплексного XML-типа данных **PROPERTY_DET_Type,** определяющий свойство Pi в момент времени t.

Характеристика x_t совпадает со словарем O_t тогда и только тогда, когда характеристика x_t может считаться экземпляром словаря O_t. Последнее означает, что:

— характеристика x_t принадлежит к одному классу C_t, скажем, классу **Class_P$_t$**;

— характеристика x_t определяется значением нескольких свойств, скажем, $P1_t$, $P2_t$,..., Pn_t;

— свойства $P1_t$, $P2_t$,..., Pn_t относятся к свойству P_t;

— свойства $P1_t$, $P2_t$,..., Pn_t – это свойства, применимые к классу **Class_P$_t$**;

Примечание 3 — $P1_t$, $P2_t$,...., Pn_t могут быть множеством всех свойств, применимых в классе **Class_P$_t$**.

— для каждого свойства $P1_t$, $P2_t$,..., Pn_t, значение, закрепленное за данным свойством, принадлежит области значений свойства в момент времени t, как это определено с помощью функции **domain$_t$**(Pi_t), i = 1 ..n..

Формально, свойство x_t будет соответствовать словарю O_t, если пользователь может кодировать его как:

$$x_t = Class\_P_t\ (P1_t = v1,\ P2_t = v2,\ ...,\ Pn_t = vn)$$

$$Class\_P_t \in O_t,\ P1_t \in O_t\ ,\ P2_t \in O_t,\ ...,\ Pn_t \in O_t$$

$$\wedge\ P1_t \in applicable\_properties_t(Class\_P_t) \wedge P2_t \in applicable\_properties_t(Class\_P_t)$$

$$\wedge\ ...\ \wedge Pn_t \in applicable\_properties_t(Class\_P_t)$$

$$\wedge\ v1 \in domain_t(P1_t) \wedge v2 \in domain_t(P2_t) \wedge ... \wedge vn \in domain_t(Pn_t)$$

Множество всех характеристик x_t, которое соответствует словарю O_t и называется «*семейством*» Pop_t в словаре O_t, определяется следующим образом:

$$Pop_t = \text{all } x_t \text{ such that } x_t \text{ conforms to } O_t$$

Мы говорим, что:

— семейство Pop_t и x_t *соответствуют* словарю O_t, и

— словарь O_t *интерпретирует* семейство Pop_t и свойство x_t.

Эти определения позволяют классифицировать различные изменения в словаре-справочнике.

Первый вид изменений в словаре – это те изменения, которые полностью не изменяют множество характеристик, которые могут определяться с помощью данного словаря, т.е. семейства словаря. Это может быть, например, в случае исправления опечаток, при введении нового перевода или при определении класса, который был переработан для уточнения его содержания без изменения смысла. В этих случаях семейство Pop_t в словаре O_t в момент времени = t становится идентичным семейству Pop_{t+1} словаря O_{t+1} в момент времени = t+1. Последнее означает, что:

— любая характеристика x_t, определенная в словаре O_t, также *соответствует* словарю O_{t+1}, поэтому словарь O_{t+1} является *обратно-совместимым* со словарем O_t, поскольку это позволяет интерпретировать все экземпляры класса; более того,

— любая характеристика x_{t+1}, определенная в словаре O_{t+1}, также *соответствует* словарю O_t, поэтому словарь O_{t+1} также является *обратно-совместимым* со словарем O_t, поскольку словарь O_t позволяет интерпретировать все экземпляры класса в словаре O_{t+1}.

В случае изменений, для которых существует прямая и обратная совместимость, нет необходимости в регистрации, если характеристика x была создана в момент времени = t или = t+1, поэтому это изменение не потребует изменения номеров версий, которые уже были закреплены за различными онтологическими понятиями в момент времени = t. Указанное изменение называется «*изменением редакции*» и будет отслеживаться путем увеличения значения XML-элемента **revision** соответствующего онтологического понятия, которое изменяется в случае изменения, влияющего на описание на нескольких языках, и/или одного или нескольких значений XML-элемента **translation_revision,** соответствующего другим языкам, на которые онтологическое понятие было переведено, если изменение оказывает влияние на описание на соответствующем языке.

Примечание 4 — Номера редакций не регистрируются в идентификаторах онтологических понятий. Характеристика, использующая онтологические понятия, будет позволять использовать словари O_t и O_{t+1} для интерпретации.

Примечание 5 — Каждое онтологическое понятие обладает XML-элементом редакции, а в случае его перевода – некоторой административной информацией, которая определяет как элемент **source_language** языка, на котором онтологическое понятие первоначально было определено, так и элемент **translation_revision** для каждого перевода.

Второй вид изменений в словаре – это те изменения, которые будут повышать его качество и позволять определять новые характеристики. Вводятся новые классы, свойства и их значения в соответствующие области значений. Для обеспечения принципа онтологической непрерывности никакой класс, свойство или значение не должны удаляться. Словарь-справочник O_{t+1}, определенный после внесения изменений, должен оставаться способным к интерпретации семейства Pop_t. Словарь O_{t+1} является еще *обратно-совместимым* со словарем O_t и позволяющим интерпретировать все экземпляры класса, однако он не будет оставаться *прямо совместимым* из-за некоторых характеристик, которые соответствуют словарю O_{t+1}, не соответствующему словарю O_t.

В случае изменений, для которых существует лишь обратная совместимость, характеристика x, которая была создана в момент времени = t+1 и зависит от измененных онтологических понятий, должна четко выражаться в своем представлении. Подобное изменение называется «*изменением версии*» и

будет отслеживаться путем увеличения значения элемента **version** измененного онтологического понятия, а также всех других онтологических понятий, которые в результате этого также были изменены.

Примечание 6 — Номера версий регистрируются в IRDI-идентификаторах онтологических понятий, а версия каждого элемента, используемого в характеристике, будет предотвращать использование словаря O_i при попытке интерпретации характеристик, основанных на конкретных версиях словаря O_{i+1}.

В таблице 3 указаны различия между редакциями и версиями словаря.

Таблица 3 – Редакции и версии словаря

	Обратная совместимость Семейство Popt *соответствует* словарю O_i.1	Прямая совместимость Семейство Popt+1 *соответствует* словарю O_i
Редакция	Да	Да
Версия	Да	Нет

10.3 Исправление ошибок

В случае возникновения ошибок в словаре-справочнике, который уже был использован для определения характеристик продукции, необходимо не только произвести исправление ошибок, но и предоставить метод, который позволит пользователям словаря понимать и производить его коррекцию. Для каждого массива данных, который содержит характеристики продукции, обработка ошибок позволит (1) выявлять, какие характеристики являются ошибочными, и (2) определить, каким образом ошибочные характеристики должны быть скорректированы, чтобы они были согласованы с откорректированным словарем.

Если ошибочные онтологические понятия еще используются для создания характеристик продукции или в замкнутой операционной среде пользователя, то характеристики могут корректироваться параллельно с корректировкой самого словаря. Ответственность за принятие решения относительно способа удаления ошибочных элементов из существующего словаря, а также способа коррекции самого словаря лежит на его поставщике.

В правилах по управлению внесением изменений в словари, определенных в настоящем стандарте, предполагается, что операционная среда пользователя является открытой и все возможные характеристики недоступны для поставщика словаря, а корректировка не может производиться вместе с корректировкой самого словаря. В подобной среде должен использоваться метод, называемый «исключение».

Исключение означает, что:

— для обеспечения обратной совместимости ошибочные онтологические понятия и/или ошибочные значения свойств будут оставаться в словаре для обеспечения этой совместимости, однако

— все ошибочные онтологические понятия связываются с XML-элементом **is_deprecated** (в значении истинно (true)), содержанием которого будет следующее: "данное онтологическое понятие или значение не должно более использоваться в новых характеристиках", и

— XML-элемент **is_deprecated_interpretation**, связанный с каждым XML-элементом **is_deprecated,** используется для определения способа, с помощью которого характеристика, связанная с исключаемым онтологическим понятием, должна изменяться для согласования с обновленным словарем.

Примечание 1 — Описание в элементе **is_deprecated_interpretation** может быть либо неформальным (для указания пользователю словаря способа обработки соответствующих данных), либо формальным – для выдачи указания компьютеру способа автоматической коррекции данных.

Примечание 2 — В существующем описании правил управления внесением изменений в словарь отсутствует указание на формальный язык описания для представления содержания элемента **is_deprecated_interpretation**, которое является задачей для коллектива, разрабатывающего CIIM-модель на этом языке.

Пример 1 — Если в классе C1 применимое к нему свойство P1 (чье значение, по предположению, выражалось в метрах), заменяется на свойство P2, которое имеет тот же смысл, но значение которого выражается в микронах, то (1) свойство P1 для XML-элемента is_deprecated устанавливается как истинное, и (2) XML-элемент is_deprecated_interpretation может устанавливаться в виде: «значение данного свойства теперь должно выражаться в микронах и регистрироваться в свойстве P2».

Пример 2 — В примере 1 значение XML-элемента is_deprecated_interpretation для свойства P1 может предоставляться, если данный подход будет согласован с сообществом пользователей словаря в виде выражения с заданным синтаксисом и представляться

значениями свойств с помощью их идентификаторов. В этом случае содержание должно выражаться как: P2:= P1 *1 000".

10.4 Правила управления внесением изменений

В данном подразделе представлены правила управления внесением изменений в словари-справочники.

10.4.1 Критерии классификации изменений

Влияние изменения онтологического понятия на семейство характеристик, которые могут интерпретироваться с помощью O_t и/или O_{t+1}, обеспечивает критерии для классификации влияния изменений на изменение версии и на устранение ошибок. В данном подразделе описан способ, с помощью которого каждое изменение должно, по крайней мере, регистрироваться в соответствии с его влиянием для гарантии того, что получатель файла обменных данных с характеристиками элементов будет восприниматься в текущем словаре как интерпретирующий этот файл или нет.

Эти правила определяют минимальные требования, однако поставщик словаря-справочника может всегда принять решение относительно обновления версии онтологического понятия, когда эти правила требовали только обновления редакции или исключения измененного элемента при обновлении редакции или версии.

Правило 1: Изменение редакции

Если после изменения понятия Ent_t (класса, типа данных, свойств ...) в словаре-справочнике O_t, (1) понятие Ent_{t+1} в новом словаре-справочнике O_{t+1} может интерпретировать все характеристики, которые могут быть определены с помощью словаря O_t, и (2) и не позволяет определять любую новую характеристику, то изменение редакции онтологического понятия будет изменяться путем изменения словарной единицы Ent_t.

Если описание измененного элемента словаря представлено только на одном языке или если оно переведено (а изменение влияет на описание на исходном языке, на котором оно было определено), то изменение должно увеличивать значение атрибута **revision** измененного элемента словаря. Если изменение понятия Ent также оказывает влияние на перевод на другие языки (на которые был переведен элемент словаря), то соответствующие переводы должны быть изменены, и эти изменения должны увеличивать значение атрибута **translation_revision** соответствующего перевода.

Пример 1 — В словаре, который доступен только на одном языке (если может интерпретироваться одно изменение определения класса без изменения характеристик), XML-элемент revision должен увеличиваться.

Пример 2 — В словаре, чьим исходным языком является английский и который переведен на немецкий и французский языки (если может интерпретироваться одно изменение определения класса на французском языке без изменения характеристик), XML-элемент translation_revision французского перевода должен увеличиваться.

Пример 3 — В словаре, чьим исходным языком является английский и который переведен на немецкий и французский языки (если может интерпретироваться одно ограничение значения XML-элемента figure без изменения характеристик класса), XML-элемент revision должен увеличиваться. Если значением элемента figure является элемент graphic_files, который не зависит от языка и поэтому применим к описаниям на всех языках, то значение элемента translation_revision на немецком и французском языках не должно увеличиваться, поскольку переведенная часть при этом не изменяется.

Пример 4 — Если кто-либо вводит «видимое» свойство в класс без того, чтобы сделать его применимым к данному классу или к любому его подклассу, то никакая характеристика не может описываться с использованием нового свойства. Никакой подлежащий изменению редакции или версии абсолютный атрибут класса не требует обновления.

Правило 2: Изменение версии

Если после изменения в словаре O_t какого-либо понятия Ent_t (класса, типа данных, свойств и т.п.) на понятие Ent_{t+1}, (1) новый словарь O_{t+1} будет способен интерпретировать все характеристики, которые могут быть определены с помощью словаря O_t, (2) но при этом и новые характеристики, которые ранее не могли интерпретироваться с помощью словаря O_t, то это изменение должно увеличивать номер версии свойства Ent_t.

Примечание 1 — Ограничительные условия оказывают влияние на те характеристики элемента, которые отвечают этим требованиям, поэтому изменение ограничительных условий должно сопровождаться увеличением номера версии класса, содержащего эти условия, однако изменение ограничительного условия не будет изменять множество характеристик, которое может интерпретироваться с помощью словаря. Таким образом, при изменении ограничительных условий на класс множество характеристик элементов, отвечающих этим ограничениям, может увеличиваться или уменьшаться без нарушения принципа онтологической неразрывности.

Пример 5 — Если кто-либо вводит в класс приемлемое свойство, то (1) все характеристики, определенные в предыдущем словаре, все еще остаются интерпретируемыми (без использования нового применимого свойства), а (2) некоторые характеристики, которые могут также определяться с помощью нового словаря, но не могут интерпретироваться с помощью предыдущего словаря (того, который использовал новое применимое свойство). По этой причине номер версии класса должен увеличиваться.

Пример 6 — Если кто-либо вводит новую альтернативную единицу в тип действительной меры какого-либо свойства, то все характеристики, определенные с помощью предыдущего словаря, могут оставаться еще интерпретируемыми (без использования новой альтернативной единицы), однако некоторые характеристики могут также определяться с помощью нового словаря, который не может интерпретироваться с помощью предыдущего словаря (путем использования новой альтернативной единицы). По этой причине номер версии класса должен увеличиваться.

Правило 3: Коррекция ошибок

Если после изменения в словаре онтологического понятия Ent_t или информационного элемента (класса, типа данных, свойства, имени, определения и т.п.) на понятие Ent_{t+1}, то новый словарь O_{t+1} не будет способен интерпретировать все характеристики, которые были определены с помощью словаря O_t. Это изменение не будет обратно совместимым и будет нарушать принцип онтологической неразрывности.

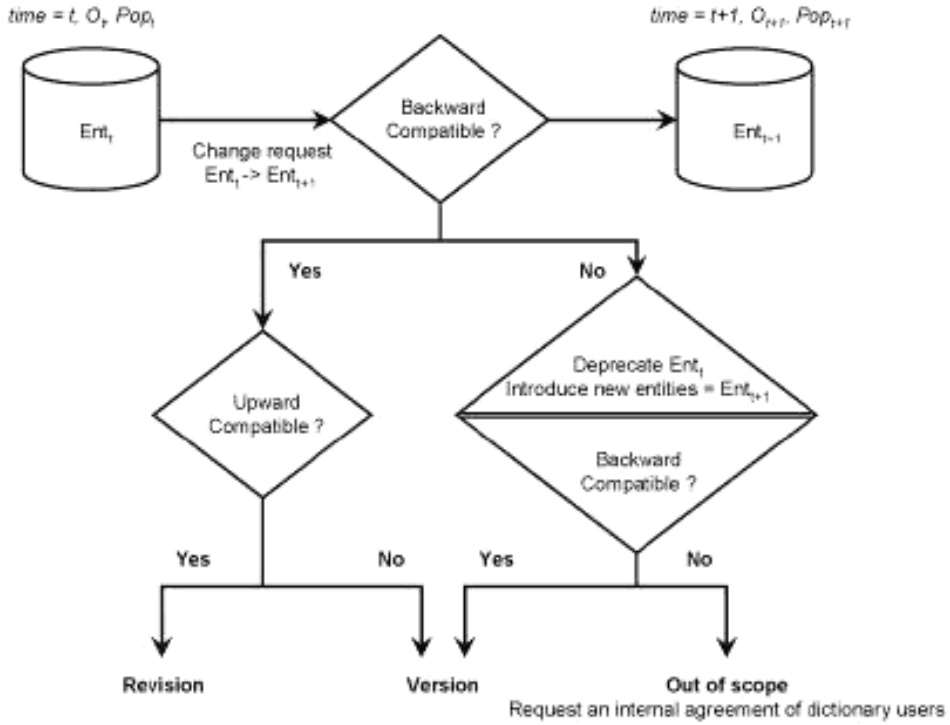
Для разрешения коррекции ошибок необходимо (1) идентифицировать те онтологические понятия, которые должны быть изменены и присвоить состояние истинности XML-элементу **is_deprecated**, (2) определить новые объекты, которые будут корректировать ошибки, и (3) описать в XML-элементе **is_deprecated_interpretation** исключаемые элементы, причину их исключения и способ изменения исключаемых элементов, чтобы они соответствовали обновленному словарю.

Пример 7 — Если кто-либо корректирует единицу в типе действительной меры какого-либо свойства, то вся продукция, характеризуемая этим свойством и зарегистрированная где-либо, должна описываться раздельно. Новый словарь при этом не будет обратно совместимым и это измерение подобным способом внесено быть не может. Таким образом, (1) соответствующее свойство должно быть исключено, (2) новое свойство (с другим идентификатором) создано, сделано «видимым» и применимым в той же области, и (3) в элементе is_deprecated_interpretation предыдущего свойства необходимо определить, что его значение должно быть, например, таким: «разделенное на 1000, а затем взятое как значение нового свойства».

Пример 8 — Если кто-либо вводит несколько новых контекстных параметров в те, от которых зависит контекстно-зависимое свойство, то все характеристики, которые содержат это свойство, должны после изменения описываться отдельно. Новый словарь не должен больше быть способным интерпретировать некоторые предыдущие характеристики, поэтому необходимо (1) исключить прежнее контекстно-зависимое свойство, (2) создать и ввести новое подобное свойство, и (3) пояснить причину исключения и, возможно (в случае, когда прежнее контекстно-зависимое свойство, по предположению, измерялось при фиксированном значении нового контекстно-зависимого параметра), то и способ преобразования исключенного контекстно-зависимого свойства в новое свойство.

Примечание 2 — В данном примере будет также возможно сохранять в словаре как прежнее, так и новое контекстно-зависимое свойство.

Рисунок 107 иллюстрирует способ классификации изменений:



Change request – Запрос на изменение; Backward compatible? – Обратно совместимое? Upward compatible? – Прямо совместимое? Deprecate Ent_t introduce new entities = Ent_{t+1} – Исключение понятия Ent_t Введение новых объектов = Ent_{t+1}; Yes – Да; No – Нет; Revision – Редакция; Version – Версия; Out of scope – Вне рассмотрения; Request an internal agreement of dictionary users – Запрос на внутреннее одобрение пользователей словаря.

Рисунок 107 — Классификация изменений в словаре

10.4.2 Взаимосвязь и распространение изменений

В словаре-справочнике каждое онтологическое понятие может существовать только в единственной версии, поэтому при увеличении номера версии этого понятия в словаре все онтологические понятия, которые относятся к данному понятию, должны быть изменены, чтобы соответствовать новой версии. В самом деле, подобное изменение должно отслеживаться на уровне идентификаторов всех ссылочных онтологических понятий для того, чтобы получить уверенность в том, что когда идентификатор онтологического понятия будет заменяться на его описание, то оно будет содержать правильные внутренние ссылки. Таким образом, каждое изменение версии онтологического понятия, упоминаемое в другом онтологическом понятии, должно представляться в виде новой версии последнего.

Принцип неизменности каждого онтологического понятия применим в рамках словаря, но не между несколькими словарями. Если класс C1 словаря D1 дает ссылку на класс C2 словаря D2 с помощью условного соотношения, то обязанностью поставщика словаря D1 является принятие решения о том, какие классы являются ссылочными, в какой версии и, возможно, о том, что свойства импортируются. Таким образом, если поставщик словаря D2 изменяет версию класса C2, то он обязан перед поставщиком словаря D1 принять решение о том, при каком условии и когда новая версия класса C2 будет упоминаться в классе C1. При этом прежняя ссылка может сохраняться, однако если номер версии ссылки увеличивается, то версия класса C1 также должна возрастать.

Эти положения указываются в последующих четырех правилах.

Правило 4: Отсутствие распространения изменений между словарями

Если класс C1 словаря-справочника D1 дает ссылку на класс C2 словаря-справочника D2 с помощью условного соотношения, и если версия класса C2 обновлена, то обязанностью поставщика словаря D1 является принятие решения о том, при каких условиях и когда класс C1 даст ссылку на новую версию класса C2. Когда это будет сделано, номер версии класса C1 должен быть увеличен.

Пример 1 — Класс C1 импортирует посредством условного соотношения свойства P1 и P2. Новое применимое свойство P3 вводится в класс C2. Поставщик словаря D1 не заинтересован в свойстве P3. Класс C1 может продолжать ссылаться на предыдущую версию класса C2.

Правило 5: Распространение версий

Увеличение номера версии любых онтологических понятий, ссылки на которые даются с помощью других онтологических понятий того же словаря, должно распространяться и на них.

Примечание 1 — Одни и те же ресурсы словаря идентифицируются одним и тем же поставщиком информации.

Примечание 2 — Когда одно онтологическое понятие ссылается на другое онтологическое понятие, то эта ссылка будет выполняться посредством IRDI-идентификатора, который включает в себя версию проектного онтологического понятия, поэтому если версия проектного понятия изменяется, то содержание исходного онтологического понятия также должно изменяться для регистрации новой (правильной) ссылки. Подобное изменение подтверждает, что новая характеристика может описываться (косвенно) с помощью исходного онтологического понятия и что его версия должна быть изменена.

Пример 2 — Изменение версии типа данных DATATYPE_Type, например, с целью расширения области его значений, будет также изменять и область значений любого свойства, которое считается связанным с этой областью для типа данных, поэтому версия этих свойств также должна обновляться. Последнее также будет изменять и множество характеристик, которые могут описываться с помощью классов, в которых они становятся применимыми с помощью XML-элемента described_element.

Пример 3 — Изменение номера версии класса приводит к изменению номера версии его субклассов и суперклассов, и т.д.

Пример 4 — Если номер версии класса определений свойства увеличивается, то номер версии этого свойства также должен увеличиваться.

Примечание 3 — Данное правило гарантирует, что если номер версии класса характеристик, используемый для описания характеристик элемента в обменном файле будет меньше или равен номеру его версии в локальном словаре на приемной системе, то эта система будет способна правильно интерпретировать эту характеристику независимо от ее сложности.

Правило 6: Расчет новых значений версии

Для каждого конкретного изменения все распространяемые изменения должны рассчитываться совместно, а номер версии каждого объекта должен увеличиваться, по крайней мере, один раз. Также допускается объединять номер различных изменений с целью расчета новых версий для множества онтологических понятий.

Правило 7: Циркуляция новой версии

Обязанностью поставщика словаря является принятие решения о том, как и когда необходимо распространять изменения.

Пример 5 — Словарь-справочник может связываться с сервером, соответствующим ИСО/ТС 29002-20, который делает доступным каждое обновление, как только оно будет утверждено.

Пример 6 — Словарь-справочник может распространяться путем выпусков. Каждый год новый выпуск объединяет в себе все обновления, полученные в течение текущего года. В этом случае все измененные онтологические понятия могут иметь только одну увеличенную версию.

10.4.3 Управление классами категорий

К классам категорий, не оказывающим влияние на характеристики элемента, не могут применяться вышеприведенные правила.

Правило 8: Подбор версий классов категорий

Увеличение версий классов категорий запрашивается в том случае, когда изменяется один или несколько их суперклассов, ссылки на которые даются с помощью XML-элемента **categorization_class_superclasses**. Все другие изменения могут регистрироваться путем увеличения номера редакции.

Изменение версий классов категорий не распространяется на классы характеристик, на которые они ссылаются. Подобные изменения регистрируются только как изменения редакции.

10.4.4 Управление версиями и редакциями словаря

В процессе обмена файлами с характеристиками элементов, основанными на словаре, нижеприведенные правила будут гарантировать, что при обмене файлами просто путем проверки версии словаря получатель файлов может знать, позволяет ли прием пригодной версии словаря интерпретировать файлы.

Правило 9: Версии и редакции словаря

После того, как обновленный словарь был распределен согласно Правилу 6:

— если номер версии любого онтологического понятия, определенного в данном словаре, увеличен, и/или если было введено новое онтологическое понятие, то номер версии словаря также должен быть увеличен;

Пример — Новое онтологическое понятие вводится в словарь тогда, когда вводится новый субкласс существующего класса или когда определяется новый «видимый» тип.

— если номер редакции любого онтологического понятия, определенный в данном словаре, увеличен, а номер версии словаря остался неизменной, то номер редакции словаря также должен быть увеличен;

— если номер версии словаря увеличен, номер его редакции должен быть сброшен на нуль.

10.5 Изменения и атрибуты словаря

10.5.1 Поддерживаемые системой атрибуты

Правила управления внесением изменений в словарь ограничиваются атрибутами, которые доступны для изменений, инициируемых по запросу пользователя. Поддерживаемые системой атрибуты, которые находятся вне области внесения изменений в словарь, поскольку они изменяются автоматически как следствие внесения другого изменения:

— Изменение редакции: если это изменение оказывает влияние на описание на исходном языке, на котором онтологическое понятие ранее было описано, то значение элемента **revision будет** увеличиваться, а значение элемента **date_of_current_revision** будет обновляться по текущей дате. Если изменение оказывает влияние на описание на одном или на нескольких других языках, на которые онтологические понятия были переведены, то значение элемента **translation_revision,** соответствующее этому языку, будет увеличиваться, а значение элемента **date_of_current_translation_revision,** соответствующее этому языку, также будет обновляться по текущей дате;

— Изменение версии: значение элемента **version** увеличивается, а значение элемента **date_of_current_version** будет обновляться с течением времени. Значение элемента **revision** будет устанавливаться на значение, определенное как максимальное для атрибута **revision,** а значение элемента **date_of_current_revision** будет обновляться по текущей дате;

— Создание нового онтологического понятия: новое онтологическое понятие создается с помощью нового элемента **code,** а значение элемента **date_of_original_definition** будет устанавливаться по текущей дате. Значение элемента **Version** будет устанавливаться на значение, определенное как минимальное для атрибута **version,** а значение элемента **date_of_current_version** будет обновляться по текущей дате. Значение элемента **Revision** будет устанавливаться на значение, определенное как минимальное для номера редакции, а значение элемента **date_of_current_revision будет** обновляться по текущей дате.

10.5.2 Атрибуты, приемлемые для внесения изменений в текст

Назначение терминологических атрибутов для таких онтологических понятий, как имена, определение, примечание, пиктограмма, состоит в пояснении того, какой вид продукции характеризуется с помощью конкретного класса словаря-справочника и какой вид характеристик продукции представляется с помощью каждого конкретного свойства.

Для гарантии обратной совместимости текстовых изменений терминологических атрибутов подобные изменения не должны сужать значения класса или свойства, даже если они могут уточнять смысл понятия, однако текстовое изменение может расширять определение класса, например, за счет принятия в расчет разработку новой продукции.

Таким образом, текстовое изменение требует изменения по крайней мере нового номера редакции, однако ответственность за принятие решения о том, должно ли это изменение приводить также и к изменению версии онтологического понятия (для гарантии того, что пользователи словаря будут иметь доступ к терминологическим атрибутам, которые будут использоваться для определения некоторых характеристик), несет поставщик словаря. Кроме того, он несет ответственность за принятие решения о том, должно ли новое онтологическое понятие связываться с новым кодом, поскольку новое определение выглядит как обратно совместимое с предыдущим определением.

10.6 Ограничительные условия на изменение словарей-справочников

В данном разделе мы подведем итог для ограничительных условий для каждого вида онтологических понятий (классов, соотношений между ними, свойств и характеристик) в процессе изменения словаря.

Неизменность классов

Существование класса характеристик не может ставиться под угрозу в процессе изменения словаря. Поскольку каждый класс характеристик допускает определение нескольких характеристик, то

любой класс, существующий в момент времени t, должен также существовать и в момент времени t', причем t' > t.

Примечание 1 — Для того, чтобы сделать модель управления внесением изменений более гибкой, классом может становиться вышедший из употребления, который затем будет помечаться как «исключенный» и, возможно, заменяться другим классом, однако он будет продолжать оставаться принадлежащим самым новым версиям словаря.

Этот принцип допускает, что недавно созданный словарь будет способен интерпретировать все ранее определенные характеристики. Ответственность за принятие решения о том, при каких условиях и до каких пор исключенные элементы будут сохраняться в каждом словаре, несет его поставщик.

Проблема неизменности классов различна для разных классов категорий. Поскольку эти классы не используются для определения характеристик, то они могут закрываться или изменяться без создания проблем с обратной совместимостью.

Неизменность свойств

Аналогично, все свойства, существующие в момент времени t, должны также существовать и в момент времени t', причем t' > t. Свойство может также становиться вышедшим из употребления, однако ни его существование, ни значение для конкретного элемента изменяться не могут. Область значений свойства может изменяться. Принимая в расчет требование обратной совместимости, эта область может только увеличиваться, а некоторые значения с течением времени могут помечаться как исключаемые.

Неизменность связи между классами и субклассами

Связь между классами и субклассами – это связь между классом и его субклассами, прямая или полученная за счет транзитивности. Эта связь поддерживает наследование свойств между суперклассом и субклассами. Требование к неизменности конкретной связи класса с субклассом между двумя классами C1 как суперклассом, и C2 – как субклассом, зависит от последствий этой связи для характеристик, определяемых с помощью субкласса:

— если субкласс C2 не наследует из класса C1 какой-либо элемент (свойство, тип, значение и т.п.), который может использоваться в характеристике, то связь типа C1-C2 может быть исключена. Последствия для номеров версии и редакции определяются с помощью правил управления внесением изменений в словарь;

— если субкласс C2 наследует из класса C1 какой-либо элемент (свойство, тип, значение и т.п.), который может использоваться в характеристике экземпляра класса C2, то связь типа C1-C2 не должна исключаться.

Отметим, что это ограничительное условие допускает большое расширение иерархии связей класса с подклассом, например, путем введения промежуточных классов между двумя классами, связанными между собой связью типа класс-подкласс.

Неизменность характеристик

Тот факт, что свойство P применимо к классу C в момент времени t, требует, чтобы свойство P оставалось применимым в этом классе в момент времени t', причем t' > t.

Примечание 2 — Последнее не требует, чтобы одни и те же применимые свойства всегда использовались для описания экземпляров одного и того же класса. Свойства, используемые для определения характеристик элемента, не зависят от расширения словаря. Они зависят в основном от требований применения словаря.

Примечание 3 — Если свойство P1 заявляется применимым к классу C2, который является субклассом класса C1, то свойство P1 может становиться применимым в классе C1 без какой бы то ни было обратной совместимости, поскольку совместимость наследуется.

Приложение А
(обязательное)

Регистрация информационных объектов

Для обеспечения однозначной идентификации информационного объекта в открытой системе настоящему стандарту присваивается следующий идентификатор объекта:

{iso standard 13584 part (32) version (1)}

Смысл этого обозначения указан в ИСО/МЭК 8824-1 и описан в ИСО 10303-1.

Приложение В

(справочное)

Компьютерно-интерпретируемые списки

В настоящем приложении содержится полная OntoML/XML-диаграмма, построенная в соответствии с различными UML-диаграммами, которые определены в настоящем стандарте. Соответствующие распечатки в компьютерно-интерпретируемой форме приведены в таблице В.1

Основная XML-диаграмма является "ontoml"- диаграммой.

К компьютерно-интерпретируемым файлам В настоящем приложении относятся следующие замечания:

Таблица В.1 – XML-диаграмма, определенная в настоящем стандарте

Наименование	ASCII-файл	Унифицированный идентификатор ресурса (URI)	Исходный документ
OntoML XML- схемы	ontoml.xsd	urn:iso:std:iso:13584:-32:ed-1 :tech:xml-schema:ontoml	ИСО 13584-32

С непосредственной или косвенной ссылкой на таблицу В.1 определяемые иным образом диаграммы приведены в таблице В.2.

Таблица В.2 – XML-диаграммы, не определенные в настоящем стандарте

Наименование	ASCII-файл	Унифицированный идентификатор ресурса (URI)	Исходный документ
Идентификатор XML- схемы	identifier.xsd	urn:iso:std:iso:ts:29002:-5:ed-1 :tech:xml-schema:identifier	ИСО/ТС 29002-5
Каталог XML- схемы	catalogue.xsd	urn:iso:std:iso:ts:29002:-10:ed-1 :tech:xml-schema:catalogue	ИСО/ТС 29002-10
Значение для XML- схемы	value.xsd	urn:iso:std:iso:ts:29002:-10:ed-1 :tech:xml-schema:value	ИСО/ТС 29002-10
Основа для XML- схемы	basic.xsd	urn:iso:std:iso:ts:29002:-4:ed-1 :tech:xml-schema:basic	ИСО/ТС 29002-4
MathML-выражение XML-схемы	mathml2.xsd	http://www.w3.org/1998/Math/MathML	www.w3.org/T R/MathML2

Приложение С

(справочное)

Требования к стандартным данным в OntoML-языке

Стандартные данные являются XML-элементами, которые должны распознаваться путем любой реализации, которая подтверждает ее принадлежность какому-либо классу соответствия в OntoML-языке.

Стандартные данные должны определяться для данной интегрированной информационной модели библиотеки для каждого класса соответствия.

С.1 Таблица для определения классов соответствия

В таблице С.1 определяются значения для XML-элементов name и application, принадлежащих комплексному типу XML-данных LIBRARY_IIM_IDENTIFICATION_Type, что допускает их использование в типе данных LIBRARY_IIM_IDENTIFICATION_Type для ссылки на OntoML-язык в любом из его классов соответствия.

Таблица С.1 – Спецификация на классы соответствия согласно ИСО 13584 LIIM 32

Класс соответствия	Обязательное обозначение имени XML-элемента типа LIBRARY_IIM_IDENTIFICATION_Type	Обязательное обозначение прикладного XML-элемента типа LIBRARY_IIM_IDENTIFICATION_Type
1	'ONTOML'	'1'
2	'ONTOML'	'2'
3	'ONTOML'	'3'
4	'ONTOML'	'4'

С.2 Стандартные данные для классов соответствия 1 - 4

Значения XML-элементов комплексного типа LIBRARY_IIM_IDENTIFICATION_Type, допустимые для использования в предоставляемом библиотечном файле, согласованном с OntoML-языком и определенным в настоящем стандарте, должны подчиняться ограничениям, принятым в данном разделе.

Неформальное ограничительное условие определяется в комплексном XML-типе данных LIBRARY_IIM_IDENTIFICATION_Type, которое должно использоваться для ссылки на классы соответствия 1 - 4 в OntoML-языке в соответствии с настоящим стандартом.

Комплексный XML-тип данных LIBRARY_IIM_IDENTIFICATION_Type допускается использовать для ссылки на классы соответствия 1 – 4 в OntoML-языке, если выполняются нижеперечисленные условия:

— XML-элемент **name** комплексного типа данных LIBRARY_IIM_IDENTIFICATION_Type, ссылающийся на интегрированную модель библиотеки LIIM 32, должен быть эквивалентен 'ONTOML', и

— XML-элемент **status** комплексного XML-типа данных LIBRARY_IIM_IDENTIFICATION_Type должен быть эквивалентен 'WD', 'CD' или 'DIS', 'FDIS', 'IS', 'TS', 'PAS' или 'ITA' , и

— XML-элемент **application** комплексного XML-типа данных LIBRARY_IIM_IDENTIFICATION_Type должен быть эквивалентен '1', '2', '3' или '4'.

Приложение D

(справочное)

Представление значений величин в ИСО 13584/МЭК 61360 и типов данных в совместно используемых в ИСО/ТС 29002-10 схемах

Для гарантии функциональной совместимости с другими стандартными представлениями продукции или определениями характеристик объекта с помощью принадлежности к классу и пар «свойство/значение», в OntoML-языке используются ресурсы функциональной совместимости, определенные в технической спецификации ИСО «Обмен основными техническими данными» серий 29002.

Более точно, OntoML-язык использует:

— XML-схему "urn:iso:std:iso:ts:29002:-5:ed-1:tech:xml-schema:identifier", определенную в ИСО/ТС 29002-5 для представления глобальных идентификаторов;

— XML-схему "urn:iso:std:iso:ts:29002:-10:ed-1:tech:xml-schema:value", определенную в ИСО/ТС 29002-10 для представления значений свойства;

— XML-схему "urn:iso:std:iso:ts:29002:-10:ed-1:tech:xml-chema:catalogue", определенную в ИСО/ТС 29002-10 для представления элементов верхнего уровня.

В настоящем приложении мы определяем, как каждый тип OntoML-данных или экземпляр объекта представляется с использованием XML-элемента, определенного в ИСО/ТС 29002-10. Поэтому в настоящем приложении представлены все обязательные ресурсы, предусматриваемые ИСО/ТС 29002-10 для конкретного использования в контексте OntoML-языка.

Как правило, аналогичные элементы ИСО/ТС 29002-10 могут использоваться для представления различных видов OntoML-экземпляров, и для каждого вида экземпляров допускается использование лишь некоторых элементов или атрибутов каждого из элементов ИСО/ТС 29002-10, поэтому должны применяться следующие правила:

— ПРАВИЛО 1: Для каждого типа OntoML-данных или экземпляра объекта мы даем его представление с помощью соответствующих элементов, определенных в ИСО/ТС 29002-10;

Пример 1 — OntoML-представление целого значения валюты согласно ИСО/ТС 29002-10 основывается на следующем XML-элементе:

```
<val:currency_value ...>
    <
</val:currency_value>
```

— ПРАВИЛО 2: В данном представлении приведены только те элементы и атрибуты, которые допускаются к использованию в данном конкретном представлении;

Пример 2 — Элементы и атрибуты, которые допускаются к использованию для OntoML-представления целого значения валюты таковы:

```
<val:currency_value currency_code="...">
    <val:integer_value>...</val:integer_value>
</val:currency_value>
```

— ПРАВИЛО 3: Значение каждого элемента и атрибута, которое допускается для использования, связывается с контентным именем, содержание которого описывается в единицах OntoML-информации;

Пример 3 — Значения каждого подэлемента и атрибута, которые допускаются для использования в OntoML-представлении целого значения валюты, таковы:

```
<val:currency_value currency_code="CurrencyCodeValue">
    <val:integer_value>IntegerValue</val:integer_value>
</val:currency_value>
```

Значение ***IntegerValue*** – это обязательное целое значение, определенное согласно разделу 3.3.17 «XML-схема, Часть 2: Типы данных». Значение ***CurencyCodeValue*** определено согласно ИСО 4217 (см. раздел 7.3.6)

— ПРАВИЛО 4: Тот факт, что это значение является вспомогательным или обязательным для конкретного типа OntoML-данных или рассматриваемого экземпляра объекта, уточняется с помощью примечания;

Пример 4 — Присвоение XML-атрибуту кода currency_code является обязательным для OntoML-представления целого значения валюты.

— ПРАВИЛО 5: Если одно и то же контентное имя используется для нескольких атрибутов или подэлементов, то оно квалифицируется с помощью имени атрибута или элемента.

В настоящем приложении определены различные представления значений согласно ИСО/ТС 29002 для соответствующих типов OntoML-данных.

В следующих подразделах согласно ИСО/ТС 29002 используются следующие префиксы к областям имен:

— **id** - для "urn:iso:std:iso:ts:29002:-5:ed-1 :tech:xml-schema:identifier";
— **val** - для "urn:iso:std:iso:ts:29002:-10:ed-1 :tech:xml-schema:value";
— **cat** - для "urn:iso:std:iso:ts:29002:-10:ed-1 :tech:xml-schema:catalogue".

D.1 Представление значения для типов данных ИСО 13584/МЭК 61360

В данном разделе определены представления значений, соответствующих различным типам данных согласно ИСО 13584/МЭК 61360, которые приведены в CIIM. Эти значения представляются в соответствии с элементом **val:schema.**

Для каждого типа OntoML-данных представление OntoML-значения определяется с помощью ИСО/ТС 29002-10.

D.1.1 Булев тип данных (boolean type)

OntoML-представление значения свойства, чья область значений является булевым (логическим) типом, выражается следующим образом:

<val:boolean_value>***BooleanValue***</val:boolean_value>

Значение ***BooleanValue*** – это булево значение, определенное согласно разделу 3.2.2 «XML-схема, Часть 2: Типы данных».

Примечание 1 — Булев тип определен в разделе 8.3.1.

Примечание 2 — Значение **val:boolean_value** определено в разделе 6.3.3 ИСО/ТС 29002-10:2009.

Примечание 3 — Присвоение содержимому XML-элемента значения **val:boolean_value** является обязательным.

Пример — Последнее справедливо для булева типа данных:

```
<val:boolean_value>true</val:boolean_value>
<val:boolean_value>1</val:boolean_value>
```

D.1.2 Строковый тип данных (string type)

OntoML-представление значения свойства, чья область значений является строковым типом, выражается следующим образом:

<val:string_value>***StringValue***</val:string_value>

Значение ***StringValue*** является строковым значением, определенным согласно разделу 3.2.1 «XML-схема, Часть 2: Типы данных».

Примечание 1 — Строковый тип определен в разделе 8.3.2.

Примечание 2 — Значение **val:string_value** определено в разделе 6.3.2 ИСО/ТС 29002-10:2009.

Примечание 3 — Присвоение содержимому XML-элемента значения **val:string_value** XML является обязательным.

Пример — Последнее справедливо для строкового типа данных:

```
<val:string_value>it is a string value</val:string_value>
<val:string_value>&#x0064;</val:string_value>
```

D.1.3 Перечень типов строковых кодов (enumeration of string codes)

OntoML-представление значения свойства, чьей областью значений является перечень типов строковых кодов, выражается следующим образом:

<val:controlled_value value_code="***StringValue***"/>

Значение ***StringValue*** является обязательным строковым значением, определенным согласно разделу 3.2.1 «XML-схема, Часть 2: Типы данных».

Примечание 1 — Перечень типов строковых кодов определен в разделе 8.3.

Примечание 2 — Значение **val:controlled_value** определено в разделе 6.8.2 ИСО/ТС 29002-10:2009.

Примечание 3 — Присвоение содержимому XML-элемента значения **value_code** является обязательным.

Пример — Последнее справедливо для перечня типов строковых кодовых значений:

```
<val:controlled_value value_code="AAA012"/>
```

D.1.4 Транслируемый строковый тип данных (translatable string type)

OntoML-представление значения свойства, чьей областью значений является транслируемый строковый тип данных, выражается следующим образом:

```
<val:localized_text_value>
    <val:content>
        <val:local_string>
            <val:content>StringValue</val:content>
            <val:language_code>LanguageCode</val:language_code>
            <val:country_code>PossibleCountryCode</val:country_code>
        </val:local_string>
        OtherPossibleLocalStrings
    </val:content>
</val:localized_text_value>
```

Значение ***StringValue*** является обязательным строковым значением, определенным согласно разделу 3.2.1 «XML-схема, Часть 2: Типы данных».

Коды ***LanguageCode*** и ***PossibleCountryCode*** являются строками, определенными согласно 8.1.1.

Строки ***OtherPossibleLocalStrings*** определяют положение, в котором могут представляться другие положения строки **val:local_string**.

Примечание 1 — Транслируемый строковый тип данных определен в разделе 8.3.2.

Примечание 2 — Значение **val:localized_text_value** определено в разделе 6.4.4 ИСО/ТС 29002-10:2009.

Примечание 3 — Код **val:language_code** определен согласно ИСО 3166-1 (см. п. 8.1.1 в настоящем стандарте).

Примечание 4 — Код **val:country_code** определен согласно ИСО 639-1 или ИСО 639-2 (см. п. 8.1.1 в настоящем стандарте).

Примечание 5 — Для каждой строки **val:local_string** должны обязательно определяться содержание **val:content** и код **val:language_code**.

Примечание 6 — При необходимости может определяться код **val:country_code**.

Примечание 7 — Для каждой пары **(val:language_code, val:country_code)** должна предоставляться только одна трансляция.

Пример — Ниже приведено значение действующего транслируемого строкового типа данных:

```
<val:localized_text_value>
    <val:content>
        <val:local_string>
            <val:content>screw</val:content>
            <val:language_code>en</val:language_code>
            <val:country_code>us</val:country_code>
        </val:local_string>
        <val:local_string>
            <val:content>vis</val:content>
            <val:language_code>fr</val:language_code>
        </val:local_string>
    </val:content>
</val:localized_text_value>
```

D.1.5 URI-тип данных (URI type)

OntoML- представление значения свойства, чьей областью значений является URI-тип данных, выражается следующим образом:

```
<val:file_value>
    <val:content>
        <val:element>
            <val:URI>URIValue</val:URI>
        </val:element>
    </val:content>
</val:file_value>
```

Значение ***URIValue*** является URI-значением, определенным согласно разделу 3.2.17 «XML-схема, Часть 2: Типы данных».

Примечание 1 — URI-тип данных определен в разделе 8.3.2.

Примечание 2 — Значение **val:file_value** определено в разделе 6.9.3 ИСО/ТС 29002-10:2009.

Примечание 3 — Никакой трансляции не должно предусматриваться.

Примечание 4 — Присвоение содержимому XML-элемента значения **val:URI** является обязательным.

Пример — Последнее справедливо для значения типа удаленного http-адреса:

```
<val:file_value>
  <val:content>
    <val:element>
      <val:URI>http://www.tc184-sc4.org/2010/OntoML</val:URI>
    </val:element>
  </val:content>
</val:file  value>
```

D.1.6 Нетранслируемый строковый тип данных (non translatable string type)

OntoML-представление значения свойства, чьей областью значений является нетранслируемый строковый тип данных, выражается следующим образом:

```
<val:string_value>StringValue</val:string_value>
```

Значение ***StringValue*** является обязательным строковым значением, определенным согласно разделу 3.2.1 «XML-схема, Часть 2: Типы данных».

Примечание 1 — Нетранслируемый строковый тип данных определен в разделе 8.3.2.

Примечание 2 — Присвоение содержимому XML-элемента значения **val:string_value** является обязательным.

Пример — Ниже приведено значение действующего нетранслируемого строкового типа данных:

```
<val:string_value>OntoML</val:string_value>
```

D.1.7 Тип данных «дата/время» (date time data type)

OntoML-представление значения свойства, чьей областью значений является тип данных «дата/время», выражается следующим образом:

```
<val:date_time_value>DateTimeValue</val:date_time_value>
```

Значение ***DateTimeValue*** является значением, определенным согласно разделу 3.2.7 «XML-схема, Часть 2: Типы данных».

Примечание 1 — Тип данных «дата/время» определен в разделе 8.3.3.

Примечание 2 — Значение **val:date_time_value** определено в разделе 6.3.7 ИСО/ТС 29002-10:2009.

Примечание 3 — Присвоение содержимому XML-элемента значения **val:boolean_value** является обязательным.

Примечание 4 — Присвоение содержимому XML-элемента значения **val:date_time_value** является обязательным.

Пример — Последнее справедливо для значения типа данных «дата/время»:

```
<val:date_time_value> 2010-03-24T20:45:00+01:00</val:date_time_value>
```

D.1.8 Тип данных «дата» (date type)

OntoML-представление значения свойства, чьей областью значений является тип данных «дата», выражается следующим образом:

```
<val:date_value>DateValue</val:date_value>
```

Значение ***DateValue*** является значением даты, определенным согласно разделу 3.2.9 «XML-схема, Часть 2: Типы данных».

Примечание 1 — Тип данных «дата» определен в разделе 8.3.3.

Примечание 2 — Значение **val:date_value** определено в разделе 6.3.6 ИСО/ТС 29002-10:2009.

Примечание 3 — Присвоение содержимому XML-элемента значения **val:date_value** является обязательным.

Пример — Последнее справедливо для значения типа «данные»:

```
<val:date_value>2008-09-30</val:date_value>
```

D.1.9 Тип данных «время» (time data type)

OntoML-представление значения свойства, чьей областью значений является тип данных «время», выражается следующим образом:

```
<val:time_value>TimeValue</val:time_value>
```

Значение ***TimeValue*** является значением времени, определенным согласно разделу 3.2.9 «XML-схема, Часть 2: Типы данных».

Примечание 1 — Тип данных «время» определен в разделе 8.3.3.

Примечание 2 — Значение **val:time_value** определено в разделе 6.3.8 ИСО/ТС 29002-10:2009.

Примечание 3 — Присвоение содержимому XML-элемента значения **val:time_value** является обязательным.

Пример — Последнее справедливо для значения типа «время»:

```
<val:time_value>20:45:00+01:00</val:time_value>
```

D.1.10 Тип данных «число» (number type)

OntoML-представление значения свойства, чьей областью значений является «число», выражается следующим образом:

```
<val:real_value>RealValue</val:real_value>
```

Значение ***RealValue*** является действительным числом, определенным согласно разделу 3.2.5 «XML-схема, Часть 2: Типы данных».

```
<val:real_value>3.21</val:real_value>
<val:real_value>12</val:real_value>
<val:real_value> 12.78e-2</val:real_value>
```

D.1.11 Тип данных «действительное число» (real type)

OntoML-представление значения свойства, чьей областью значений является «действительное число», выражается следующим образом:

```
<val:real_value>RealValue</val:real_value>
```

Значение ***RealValue*** является действительным числом, определенным согласно разделу 3.2.5 «XML-схема, Часть 2: Типы данных».

Примечание 1 — Тип данных «действительное число» определен в разделе 8.3.5.

Примечание 2 — Значение **val:real_value** определено в разделе 6.3.10 ИСО/ТС 29002-10:2009.

Примечание 3 — Присвоение содержимому XML-элемента значения **val:real_value** является обязательным.

Пример — Последнее справедливо для значения типа «действительное число»:

```
<val:real_value>3.21</val:real_value>
<val:real_value>12</val:real_value>
<val:real_value> 12.78e-2</val:real_value>
```

D.1.12 Тип данных «целое число» (integer type)

OntoML-представление значения свойства, чьей областью значений является «целое число», выражается следующим образом:

```
<val:integer_value>IntegerValue</val:integer_value>
```

Значение ***IntegerValue*** является целым значением, определенным согласно разделу 3.3.17 «XML-схемы, Часть 2: Типы данных».

Примечание 1 — Тип данных «целое число» определен в разделе 8.3.5.

Примечание 2 — Значение **val:integer_value** определено в разделе 6.3.13 ИСО/ТС 29002-10:2009.

Примечание 3 — Присвоение содержимому XML-элемента значения **val:integer_value** является обязательным.

Пример — Последнее справедливо для значения типа «целое число»:

```
<val:integer_value>10</val:integer_value>
```

D.1.13 Тип данных «рациональное число» (rational type)

OntoML-представление значения свойства, чьей областью значений является тип «рациональное число», выражается следующим образом:

```
<val:rational_value>
    <val:whole_part>WholePartIntegerValue</val:whole_part>
    <val:numerator>NumeratorIntegerValue</val:numerator>
    <val:denominator>DenominatorIntegerValue</val:denominator>
</val:rational_value>
```

Значения ***WholePartIntegerValue***, ***NumeratorIntegerValue*** и ***DenominatorIntegerValue*** являются целыми числами, определенными согласно разделу 3.3.17 «XML-схема, Часть 2: Типы данных».

Примечание 1 — Значение ***DenominatorIntegerValue*** должно быть ненулевым целым числом.

Примечание 2 — Тип данных «рациональное число» определен в разделе 8.3.5.

Примечание 3 — Значение **val:rational_value** определено в разделе 6.3.13 ИСО/ТС 29002-10:2009.

Примечание 4 — Присвоение содержимому XML-элемента значения **val:whole_part** является необязательным.

Примечание 5 — Присвоение содержимым XML-элементов значений **val:numerator** и **val:denominator** является обязательным.

Пример — Последнее справедливо для значения типа «рациональное число»:

```
<val:rational_value>
    <val:whole_part>5</val:whole_part>
    <val:numerator>2</val:numerator>
    <val:denominator>3</val:denominator>
</val:rational_value>
```

D.1.14 Тип данных «действительное значение валюты» (real currency type)

OntoML-представление значения свойства, чьей областью значений является «действительное значение валюты», выражается следующим образом:

```
<val:currency_value currency_code="CurrencyCodeValue" currency_ref="CurrencyIRDI">
    <val:real_value>RealValue</val:real_value>
</val:currency_value>
```

Значение ***RealValue*** является действительным числом, определенным согласно разделу 3.2.5 «XML-схема, Часть 2: Типы данных».

Значение ***CurrencyCodeValue*** определяется согласно ИСО 4217 (см. раздел 8.3.6 в настоящем стандарте).

Значение ***CurrencyIRDI*** – это возможная ссылка на валюту, определенную в каталоге валют.

Примечание 1 — Тип данных «действительное значение валюты» определен в разделе 8.3.6.

Примечание 2 — Значение **val:currency_value** определено в разделе 6.3.3 ИСО/ТС 29002-10:2009.

Примечание 3 — Присвоение XML-атрибуту значения **currency_code** является необязательным. Оно предоставляется только в том случае, когда оно не определено в связанном типе «действительное значение валюты».

Примечание 4 — Идентификатор **currency_ref** является действующим идентификатором модуля (IRDI), как это определено в разделе 9.1.3.3.

Примечание 5 — Присвоение XML-атрибуту значения **currency_ref** является необязательным.

Примечание 6 — Если предоставляются оба XML-атрибута **currency_code** и **currency_ref**, то они должны обозначать одни и те же данные о валюте.

Примечание 7 — Присвоение содержимому XML-элемента значения **val:real_value** является обязательным.

Пример — Последнее справедливо для значения типа «действительное значение валюты»:

```
<val:currency_value currency_code="EUR">
        <val:real_value>10.53</val:real_value>
</val:currency_value>
```

D.1.15 Тип данных «целое значение валюты» (integer currency type)

OntoML-представление значения свойства, чьей областью значений является «целое значение валюты», выражается следующим образом:

```
<val:currency_value currency_code="CurrencyCodeValue" currency_ref="CurrencyIRDI">
        <val:integer_value>IntegerValue</val:integer_value>
</val:currency_value>
```

Значение ***IntegerValue*** является обязательным целым значением, определенным согласно разделу 3.3.17 «XML-схема, Часть 2: Типы данных».

Значение ***CurrencyCodeValue*** определяется согласно ИСО 4217 (см. раздел 8.3.6 в данной части ИСО 13584).

Значение ***CurrencyIRDI*** – это возможная ссылка на валюту, определенную в каталоге валют.

Примечание 1 — Тип данных «целое значение валюты» определен в разделе 8.3.6.

Примечание 2 — Значение **val:currency_value** определено в разделе 6.3.3 ИСО/ТС 29002-10:2009.

Примечание 3 — Присвоение XML-атрибуту значения **currency_code** является необязательным. Оно предоставляется только в том случае, когда оно не определено в связанном типе «целое значение валюты».

Примечание 4 — Идентификатор **currency_ref** является действующим идентификатором единицы (IRDI), как это определено в разделе 9.1.3.3.

Примечание 5 — Присвоение XML-атрибуту значения **currency_ref** является необязательным.

Примечание 6 — Если предоставляются оба XML-атрибута **currency_code** и **currency_ref**, то они должны обозначать одни и те же понятия о валюте.

Примечание 7 — Присвоение содержимому XML-элемента значения **val: integer_value** является обязательным.

Пример — Последнее справедливо для значения типа «целое значение валюты»:

```
<val:currency_value currency_code="EUR">
        <val:integer_value>10</val:integer_value>
</val:currency_value>
```

D.1.16 Тип данных «действительная мера» (real measure type)

OntoML-представление значения свойства, чьей областью значений является тип «действительная мера», выражается следующим образом:

```
<val:measure_single_number_value UOM_code="UnitValue" UOM_ref="UnitIRDI">
        <val:real_value>RealValue</val:real_value>
</val:measure_single_number_value>
```

Значение ***RealValue*** является обязательным действительным значением, определенным согласно разделу 3.2.5 «XML-схема, Часть 2: Типы данных».

Значение ***UnitValue*** является возможной единицей, принадлежащей перечню единиц, определенному в соответствующей спецификации на тип «действительная мера» (математическое строковое представление основной единицы или альтернативных единиц, см. раздел 8.3.7).

Примечание 1 — При необходимости используйте ИСО 843.

Значение ***UnitIRDI*** является возможной ссылкой на единицу, определенную в каталоге единиц.

Примечание 2 — Идентификатор **UOM_ref** является действующим идентификатором единицы (IRDI), как это определено в разделе 9.1.3.3.

Примечание 3 — Тип «действительная мера» определен в разделе 8.3.7.

Примечание 4 — Значение **val:measure_single_number_value** определено в разделе 6.5.5 ИСО/ТС 29002-10:2009.

Примечание 5 — Необходимо производить присвоение XML-атрибуту значения **UOM_code** или **UOM_ref**. Если предоставляются оба атрибута **UOM_ref** и **UOM_code**, то они должны обозначать одни и те же понятия о единице.

Примечание 6 — Присвоение XML-атрибуту значения **val:real_value** является обязательным.

Пример — Последнее справедливо для значения типа «действительная мера»:

```
<val:measure_single_number_value UOM_code="mm">
    <val:real_value>10.53</val:real_value>
</val:measure_single_number_value>
```

D.1.17 Тип данных «целая мера» (integer measure type)

OntoML-представление значения свойства, чьей областью значений является тип «целая мера», выражается следующим образом:

```
<val:measure_single_number_value UOM_code="UnitValue" UOM_ref="UnitIRDI">
    <val:integer_value>IntegerValue</val:integer_value>
</val:measure_single_number_value>
```

Значение ***IntegerValue*** является обязательным целым значением, определенным согласно разделу 3.3.17 «XML-схема, Часть 2: Типы данных».

Значение ***UnitValue*** является возможной единицей, принадлежащей перечню единиц, определенному в соответствующей спецификации на тип «целая мера» (математическое строковое представление основной единицы или альтернативных единиц, см. раздел 8.3.7).

Значение ***UnitIRDI*** является возможной ссылкой на единицу, определенную в каталоге единиц.

Примечание 1 — При необходимости используйте ИСО 843.

Примечание 2 — Идентификатор **UOM_ref** является действующим идентификатором единицы (IRDI), как это определено в разделе 9.1.3.3.

Примечание 3 — Тип «целая мера» определен в разделе 8.3.7.

Примечание 4 — Значение **val:measure_single_number_value** определено в разделе 6.5.5 ИСО/ТС 29002-10:2009.

Примечание 5 — Необходимо производить присвоение XML-атрибуту значения **UOM_code** или **UOM_ref**. Если предоставляются оба атрибута **UOM_ref** и **UOM_code**, то они должны обозначать одни и те же понятия о единице.

Примечание 6 — Присвоение XML-атрибуту значения **val:integer_value** является обязательным.

Пример — Последнее справедливо для значения типа «целая мера»:

```
<val:measure_single_number_value UOM_code="mm">
    <val:integer_value>5</val:integer_value>
</val:measure_single_number_value>
```

D.1.18 Тип данных «рациональная мера» (rational measure type)

OntoML-представление значения свойства, чьей областью значений является тип «рациональная мера», выражается следующим образом:

```
<val:measure_single_number_value UOM_code="UnitValue" UOM_ref="UnitIRDI">
    <val:rational_value>
        <val:whole_part>WholePartIntegerValue</val:whole_part>
        <val:numerator>NumeratorIntegerValue</val:numerator>
        <val:denominator>DenominatorIntegerValue</val:denominator>
    </val:rational_value>
</val:measure_single_number_value>
```

Значения ***WholePartIntegerValue, NumeratorIntegerValue*** и ***DenominatorIntegerValue*** являются целыми числами, определенными согласно разделу 3.3.17 «XML-схема, Часть 2: Типы данных».

Значение ***UnitValue*** является возможной единицей, принадлежащей перечню единиц, определенному в соответствующей спецификации на тип «действительная мера» (математическое строковое представление основной единицы или альтернативных единиц, см. раздел 8.3.7).

Значение ***UnitIRDI*** является возможной ссылкой на единицу, определенную в каталоге единиц.

Примечание 1 — При необходимости используйте ИСО 843.

Примечание 2 — Значение ***DenominatorIntegerValue*** должно быть ненулевым целым числом.

Примечание 3 — Атрибут **UOM_ref** является действующим IRDI-идентификатором, определенным в разделе 9.1.3.3.

Примечание 4 — Тип «рациональная мера» определен в разделе 8.3.7.

Примечание 5 — Значение **val:measure_single_number_value** определено в разделе 6.5.5 ИСО/ТС 29002-10:2009.

Примечание 6 — Необходимо производить присвоение XML-атрибуту значения **UOM_code** или **UOM_ref**. Если предоставляются оба атрибута **UOM_ref** и **UOM_code**, то они должны обозначать одни и те же понятия о единице.

Примечание 7 — Присвоение содержанию XML-элемента значения **val:whole_part** является необязательным.

Примечание 8 — Присвоение содержанию XML-элементов значений **valnumerator** и **val:denominator** является обязательным.

Пример — Последнее справедливо для значения типа «рациональная мера»:

```
<val:measure_single_number_value UOM_code="mm">
    <val:rational_value>
        <val:whole_part>5</val:whole_part>
        <val:numerator>2</val:numerator>
        <val:denominator>3</val:denominator>
    </val:rational_value>
</val:measure_single_number_value>
```

D.1.19 Тип данных «перечень целочисленных кодов» (enumeration of integer codes)

OntoML-представление значения свойства, чьей областью значений является перечень типов целочисленных кодов, выражается следующим образом:

```
<val:controlled_value value_code="StringValue"/>
```

Значение ***String value*** должно означать целое число.

Примечание 1 — Перечень типов целочисленных кодов определен в разделе 8.3.8.

Примечание 2 — Значение **val:controlled_value** определено в разделе 6.8.2 ИСО/ТС 29002-10:2009.

Примечание 3 — Присвоение XML-атрибуту значения **value_code** является обязательным.

Пример — Последнее справедливо для перечня типов целочисленных кодов:

```
<val:controlled_value value_code="38"/>
```

D.1.20 Тип данных «пакет» (bag type)

OntoML-представление значения свойства, чьей областью значений является тип данных «пакет», выражается следующим образом:

```
<val:bag_value>
    ABagOfBaseTypeValue
</val:bag_value>
```

Значение ***ABagOfBaseTypeValue*** является любым видом числа, определенным в настоящем приложении.

Примечание 1 — Любое значение, определенное в типе данных «пакет», должно принадлежать одному и тому же основному типу.

Примечание 2 — Допускается использование дублированных значений.

Примечание 3 — Тип данных «пакет» определен в разделе 8.3.9.1.

Примечание 4 — Значение **val:bag_value** определено в разделе 6.7.5 ИСО/ТС 29002-10:2009.

Примечание 5 — Значение **val:bag_value** с содержимым ***ABagOfBaseTypeValue*** является обязательным.

Пример — Ниже приведено действующее значение типа данных «пакет», для которого составные элементы являются целочисленными:

```
<val:bag_value>
    <val:integer_value>10</val:integer_value>
    <val:integer_value>10</val:integer_value>
    <val:integer_value>30</val:integer_value>
</val:bag_value>
```

D.1.21 Тип данных «набор» (set type)

OntoML-представление значения свойства, чьей областью значений является тип данных «набор», выражается следующим образом:

```
<val:set_value>
        ASetOfBaseTypeValue
</val:set_value>
```

Значение ***ASetOfBaseTypeValue*** является любым видом числа, определенным в настоящем приложении.

Примечание 1 — Любое значение, определенное в типе данных «набор», должно принадлежать одному и тому же основному типу.

Примечание 2 — Допускается использование дублированных значений.

Примечание 3 — Тип данных «набор» определен в разделе 8.3.9.2.

Примечание 4 — Значение **val:set_value** определено в разделе 6.7.4 ИСО/ТС 29002-10:2009.

Примечание 5 — Значение **val:set_value** с содержимым (***ASetOfBaseTypeValue***) является обязательным.

Пример — Ниже приведено действующее значение типа данных «набор», для которого составные элементы являются целочисленными:

```
<val:set_value>
    <val:integer_value>10</val:integer_value>
    <val:integer_value>20</val:integer_value>
    <val:integer_value>30</val:integer_value>
</val:set_value>
```

D.1.22 Тип данных «список» (list type)

OntoML-представление значения свойства, чьей областью значений является тип данных «список», выражается следующим образом:

```
<val:sequence_value>
        AListOfBaseTypeValue
</val: sequence _value>
```

Значение ***AListOfBaseTypeValue*** является любым видом числа, определенным в настоящем приложении.

Примечание 1 — Любое значение, определенное в типе данных «список», должно принадлежать одному и тому же основному типу.

Примечание 2 — В зависимости от типа списка дублированные значения могут допускаться или не допускаться.

Примечание 3 — Тип данных «список» определен в разделе 8.3.9.3.

Примечание 4 — Значение **val:sequence_value** определено в разделе 6.7.6 ИСО/ТС 29002-10:2009.

Примечание 5 — Значение **val:sequence_value** с содержимым ***AListOfBaseTypeValue*** является обязательным.

Пример — Ниже приведено действующее значение типа данных «список», для которого составные элементы являются целочисленными:

```
<val:sequence_value>
    <val:integer_value>10</val:integer_value>
    <val:integer_value>20</val:integer_value>
    <val:integer_value>30</val:integer_value>
</val: sequence _value>
```

D.1.23 Тип данных «массив» (array type)

OntoML-представление значения свойства, чьей областью значений является тип данных «массив», выражается следующим образом:

```
<val:sequence_value>
        AnArrayOfBaseTypeValue
</val: sequence_value>
```

Значение ***AnArrayOfBaseTypeValue*** является любым видом числа, определенным в настоящем

приложении.

Примечание 1 — Любое значение, определенное в типе данных «пакет», должно принадлежать одному и тому же основному типу или может быть нулевым значением.

Примечание 2 — В зависимости от спецификации на тип данных «массив» дублированные значения могут допускаться или не допускаться.

Примечание 3 — Тип данных «массив» определен в разделе 8.3.9.1.4.

Примечание 4 — Значение **val:sequence_value** определено в разделе 6.7.6 ИСО/ТС 29002-10:2009.

Примечание 5 — Значение **val:sequence_value** с содержимым (***AnArrayOfBaseTypeValue***) является обязательным.

Пример — Ниже приведено действующее значение типа данных «массив», для которого второй элемент не определен (нулевое значение):

```
<val:sequence_value>
    <val:integer_value>10</val:integer_value>
    <val:null_value/>
    <val:integer_value>30</val:integer_value>
</val: sequence_value>
```

D.1.24 Множество с подмножеством данных типа «ограничительное условие» (set with subset constraint type)

OntoML-представление значения свойства, чьей областью значений является множество данных с подмножеством типа «ограничительное условие», выражается следующим образом:

```
<val:set_value>
    ASetOfBaseTypeValue
</val:set_value>
```

Значение ***ASetOfBaseTypeValue*** является любым видом числа, определенным в настоящем приложении.

Примечание 1 — Любое значение, определенное в типе данных «массив», должно принадлежать одному и тому же основному типу.

Примечание 2 — Набор данных с поднабором типа «ограничительное условие» определен в разделе 8.3.9.5.

Примечание 3 — Элемент **val:set_value** определен в разделе 6.7.4 ИСО/ТС 29002-10:2009.

Примечание 4 — Динамические граничные значения для набора данных с поднабором типа «ограничительное условие», используемые для перекрытия статических граничных условий и определенные на уровне типа, не могут быть преобразованы.

Примечание 5 — Значение **val:set_value** с содержимым (***ASetOfBaseTypeValue***) является обязательным.

Пример — Ниже приведен действующий набор данных с поднабором типа «ограничительное условие», для которого составные элементы являются целочисленными:

```
<val:set_value>
    <val:integer_value>10</val:integer_value>
    <val:integer_value>20</val:integer_value>
    <val:integer_value>30</val:integer_value>
</val:set_value>
```

D.1.25 Тип данных «ссылка на класс» (class reference type)

OntoML-представление значения свойства, чьей областью значений является тип «ссылка на класс», основанный на методе прямой ссылки, выражается следующим образом:

```
<val:item_reference_value item_local_ref="LocalIdRef"/>

<cat:item class_ref="ClassIRDI" local_id="LocalId">
    <cat:property_value property_ref="PropertyIRDI">
    ...
    </cat:property_value>
...
</cat:item>
```

Значение ***Local Id*** относится к локальному XML-идентификатору и предназначено для использования в качестве метки на значение свойства, чьей областью значений является тип «ссылка на класс». Ссылка определяется путем установления значения ***LocalIdRef*** XML-атрибута **item_local_ref**. Значение ***LocalId*** определено в соответствии с разделом 3.3.8 «XML-диаграмма, Часть 2: Типы данных».

Примечание 1 — Значения ***LocalId/LocalIdRef*** аналогичны используемым в ссылочном методе ID/IDREF XML.

Значение ***LocalIdRef*** определяется в соответствии с разделом 3.3.6 XML-схемы (часть 2: Типы данных).

Значение ***ClassIRDI*** – это IRDI-идентификатор действующего понятия класса, определенный в соответствии с разделом 9.1.3.1.

Значение ***PropertyIRDI*** - это IRDI-идентификатор действующего понятия класса, определенный в соответствии с разделом 9.1.3.2.

Примечание 2 — Элемент **cat:property_value** представляет собой конструкцию пары «свойство-значение» (см. раздел D.2.1.4).

Примечание 3 — Тип ссылочного класса определен в разделе 8.3.10.

Примечание 4 — Присвоение XML-атрибутов **item_local_ref** и **local_id** является обязательным.

Примечание 5 — Присвоение XML-атрибута **class_ref** является обязательным.

Примечание 6 — Элемент **val:item_reference_value** определен в разделе 6.9.2 ИСО/ТС 29002-10:2009.

Примечание 7 — Элемент **cat:item** определен в разделе D.2.2.1 настоящего приложения.

Пример — Далее приведены значения действующего типа ссылочного класса: ссылочный элемент определяется в соответствии с классом, идентифицируемым с помощью IRDI-идентификатора "0123-ABC#01-SCREW#1" , и локально идентифицируется (XML-элемент local_id) с помощью локального идентификатора "ITEM_ID_1". Ссылка (XML-элемент item_local_ref) устанавливается посредством этого локального идентификатора.

```
<val:item_reference_value item_local_ref="ITEM_ID_1"/>
...
<cat:item class_ref="0123-ABC#01-SCREW#1" local_id="ITEM_ID_1">
    <cat:property_value property_ref="0123-ABC#02-DIAMETER#1">
    ...
    </cat:property_value>
...
</cat:item>
```

D.1.26 Тип данных «уровень» (level type)

OntoML-представление значения свойства, чьей областью значений является тип данных «уровень», выражается следующим образом:

```
<val:measure_qualified_number_value UOM_code="UnitValue" UOM_ref="UnitIRDI">
    <val:qualified_value qualifier_code="Qualifier">
        NumericValueOrNullValue
    </val:qualified_value>
    OtherPossibleQualifiedValues
</val:measure_qualified_number_value>
```

Значение ***UnitValue*** является возможной единицей, которая принадлежит списку единиц, определенному в соответствующем описании типов действительной меры (представление в виде математической строки основной единицы или одной из ее альтернативных единиц, см. раздел 8.3.7).

Значение ***UnitIRDI*** – это возможная ссылка на IRDI-идентификатор единицы, определенной в словаре единиц.

Значение ***Qualifier*** означает действующий OntoML-квалификатор, который может принимать следующие значения: **min**, **nom**, **max** или **typ**.

Значение ***NumericValueOrNullValue*** является либо действительным значением (см. раздел D.1.11), целым значением (см. раздел D.1.12) или нулевым значением (см. раздел 7.3.11).

Значение ***OtherPossibleQualifiedValues*** относится ко всем другим возможным подходящим значениям (XML-компонент **val:qualified_value**), определенным в описании связанного типа уровня.

Примечание 1 — Присвоение XML-атрибута **qualifier_code** является обязательным.

Примечание 2 — Необходимо присвоить XML-атрибут **UOM_code** или **UOM_ref**. Если присвоены оба XML-атрибута **UOM_ref** и **UOM_code**, то они должны обозначать одно и то же понятие единицы.

Примечание 3 — XML-атрибут **UOM_ref** является действующим IRDI-идентификатором (см. раздел 9.1.3.3).

Примечание 4 — Любое соответствующее значение должно быть типизировано с использованием того же основного типа, или в случае, когда оно отсутствует, значение должно представляться как нулевое.

Примечание 5 — Тип данных «уровень» определен в разделе 8.3.11.

Примечание 6 — Элемент **val:measure_qualified_number_value** определен в разделе 6.5.3 ИСО/ТС 29002-10:2009.

Пример — Далее приведен пример описания действующего значения: Существует физическая мера, чьей единицей (XML-атрибут UOM_code) является миллиметр (mm). Эта физическая мера связывается с двумя квалификаторами (XML-атрибут qualifier_code): минимальным целым значением (min) и типовым значением (typ), которые отсутствуют и затем представляются с помощью нулевого значения согласно ИСО/ТС 29002 (XML-элемент null_value).

```
<val:measure_qualified_number_value UOM_code="mm">
    <val:qualified_value qualifier_code="min">
        <val:integer_value>10</val:integer_value>
    </val:qualified_value>
    <val:qualified_value qualifier_code="typ">
        <val:null_value/>
    </val:qualified_value>
</val:measure_qualified_number_value>
```

D.1.27 Тип именованных данных

OntoML-представление значения свойства, чьей областью значений является тип именованных данных, выражается в соответствии с основополагающим типом именованных данных.

D.2 Представление элементов

В данном разделе определено представление значений элементов (т.е. экземпляров) в OntoML-языке. В соответствии с ИСО/ТС 29002, эти экземпляры выражаются с использованием ссылки на класс, где приведено его описание, и с использованием множества значений свойств, каждое из которых определяется путем ссылки на свойство и типизированное значение.

Ссылки на классы и свойства представляются с помощью глобального идентификатора, как это определено в разделах 9.1.3.1 и 9.1.3.2. Типизированные значения определяются в разделе D.1 настоящего приложения.

В данном разделе вначале приводится представление (согласно ИСО/ТС 29002-10) значений OntoML-свойства, которое зависит от его вида (независимое свойство, условное свойство, зависимое свойство и экземплярное свойство). После этого в соответствии с ИСО/ТС 29002-10 определяется выражение OntoML-экземпляров класса (класса элементов или класса функциональных моделей).

D.2.1 Представление пары «свойство-значение»

В данном подразделе определены OntoML-представления для пар:

- «независимое свойство - значение»;
- «условное свойство - значение»;
- «зависимое свойство - значение»;
- «свойство экземпляра - значение».

D.2.1.1 Независимое свойство

OntoML-представление значения свойства, которое определяется как независимое, выражается следующим образом:

```
<cat:property_value property_ref="PropertyIRDI" subitem_path_property_ref="PropertyIRDIs">
    ValueRepresentation
</cat:property_alue>
```

Значение ***PropertyIRDI*** – это действующий IRDI-идентификатор понятия свойства, определенный согласно разделу 9.1.3.2.

Примечание 1 — Присвоение XML-атрибута **property_ref** является обязательным.

Значение ***PropertyIRDIs*** – это список IRDI-идентификаторов понятия свойства согласно разделу 9.1.3.2, который позволяет описывать путь от исходного свойства (определяемого с помощью XML-атрибута **property_ref**), типом данных которого является тип ссылочного класса, до целевого свойства, который участвует в описании вводимого суб-экземпляра класса того экземпляра, для которого исходное

свойство определено.

Примечание 2 — Присвоение XML-атрибута **subitem_path_property_ref** является необязательным. Оно может использоваться только для идентифицированного свойства (XML-атрибут **property_ref**), для которого основополагающим типом данных является тип ссылочного класса (см. раздел 8.3.10).

Примечание 3 — XML-атрибут **subitem_path_property_ref** может использоваться как простой способ представления введенной пары «свойство-значение».

Значение ***ValueRepresentation*** (обязательное) относится к любому виду типизированного значения, определенного в настоящем приложении.

Примечание 4 — Независимое свойство определено в разделе 6.7.4.

Примечание 5 — Элемент **cat:property_value** определен в разделе 5.2.4 ИСО/ТС 29002-10:2009.

Примечание 6 — Присвоение XML-элемента **cat:property_value** является обязательным.

Пример 1 — Нижеследующее относится к действующему представлению значения независимого свойства: это свойство идентифицируется с помощью IRDI-идентификатора "0123-ABC#02-DIAMETER#1", а связанное значение – это значение целой меры (см. раздел D.1.17).

```
<cat:property_value property_ref="0123-ABC#02-DIAMETER#1">
    <val:measure_single_number_value UOM_code="mm">
        <val:integer_value>5</val:integer_value>
    </val:measure_single_number_value>
</cat:property_value>
```

Пример 2 — Предположим, что ноутбук описывается с помощью независимого свойства, называемого its mother board, чьим идентификатором является IRDI-идентификатор "0123-ABC#02-ITS_MOTHERBOARD#1" и чьим типом данных является класс mother board, который сам по себе идентифицируется с помощью IRDI-идентификатора "0123-ABC#01-MOTHERBOARD#1". Предположим также, что этот класс mother board описывается с помощью независимого свойства, называемого its processor unit, чьим идентификатором является IRDI-идентификатор "0123-ABC#02-ITS_PROCESSOR_UNIT#1", и чьим типом данных является класс processor unit, который сам по себе идентифицируется с помощью IRDI-идентификатора "0123-ABC#01-PROCESSORUNIT#1". Кроме того, предположим, что класс processor unit описывается с помощью независимого свойства, называемого its processor, чьим идентификатором является IRDI-идентификатор "0123-ABC#02-ITS_PROCESSOR#1", и чьим типом данных является класс processor, который сам по себе идентифицируется с помощью IRDI-идентификатора "0123-ABC#01-PROCESSOR#1". Наконец, предположим, что класс processor описывается с помощью независимого свойства, называемого frequency, чьим идентификатором является IRDI-идентификатор "0123-ABC#02-FREQUENCY#1", и чьим типом данных является тип действительной меры. Ниже приведено представление описанного сочетания независимых свойств:

```
<cat:property_value property_ref="0123-ABC#02-ITS_MOTHERBOARD#1" subitem_path_property_ref="0123-ABC#02-
ITS_PROCESSOR_UNIT#1 0123-ABC#02-ITS_PROCESSOR#1 0123-ABC#02-FREQUENCY#1">
    <val:measure_single_number_value UOM_code="GHz">
        <val:real_value>4.2</val:real_value>
    </val:measure_single_number_value>
</cat:property_value>
```

В другом варианте и в соответствии с OntoML-представлением свойства, чьей областью значений является тип ссылочного класса (см. раздел D.1.25), тот же пример может быть представлен и в следующем виде:

```
<cat:item class_ref="10123-ABC#01-PROCESSOR#1" local_id="PROCESSOR_ID">
    <cat:property_value property_ref="0123-ABC#02-FREQUENCY#1">
        <val:measure_single_number_value UOM_code="GHz">
            <val:real_value>4.2</val:real_value>
        </val:measure_single_number_value>
    </cat:property_value>
</cat:item>
```

D.2.1.2 Условное свойство

OntoML-представление значения свойства, которое определяется как условное, выражается следующим образом:

```
<cat:property_value property_ref="PropertyIRDI" subitem_path_property_ref="PropertyIRDIs">
        ValueRepresentation
</cat:property_alue>
```

Значение ***PropertyIRDI*** – это действующий IRDI-идентификатор понятия свойства, определенный согласно разделу 9.1.3.2.

Примечание 1 — Присвоение XML-атрибута **property_ref** является обязательным.

Значение ***PropertyIRDIs*** – это список IRDI-идентификаторов понятия свойства согласно разделу 9.1.3.2, который позволяет описывать путь от исходного свойства (определяемого с помощью XML-атрибута **property_ref**), типом данных которого является тип ссылочного класса, до целевого свойства, который участвует в описании вводимого суб-экземпляра класса того экземпляра, для которого исходное свойство определено.

Примечание 2 — Присвоение XML-атрибута **subitem_path_property_ref** является необязательным. Он может использоваться только для идентифицированного свойства (XML-атрибут **property_ref**), для которого основополагающим типом данных является тип ссылочного класса (см. раздел 8.3.10).

Примечание 3 — XML-атрибут **subitem_path_property_ref** может использоваться как простой способ представления введенной пары «свойство-значение» (см. пример в разделе D.2.1.1).

Значение ***ValueRepresentation*** (обязательное) относится к любому виду типизированного значения, определенному в настоящем приложении.

Примечание 4 — Условное свойство определено в разделе 6.7.4.

Примечание 5 — Элемент **cat:property_value** определен в разделе 5.2.4 ИСО/ТС 29002-10:2009.

Примечание 6 — Присвоение XML-элемента **cat:property_value** является обязательным.

Пример — Нижеследующее относится к представлению значения условного свойства: это свойство идентифицируется с помощью IRDI-идентификатора "0123-ABC#02-LOAD#1", а связанное значение – это значение действительной меры (см. раздел D.1.16).

```
<cat:property_value property_ref="0123-ABC#02-LOAD#1">
        <val:measure_single_number_value UOM_code="N">
                <val:real_value>1512.37</val:real_value>
        </val:measure_single_number_value>
</cat:property_value>
```

D.2.1.3 Зависимое свойство

OntoML-представление значения свойства, которое определяется как зависимое, выражается следующим образом:

```
<cat:property_value property_ref="PropertyIRDI" subitem_path_property_ref="PropertyIRDIs">
        DependentValueRepresentation
        <val:environment>
                <val:property_value property_ref= PropertyIRDI">
                        ConditionValueRepresentation
                </val:property_value>
                OtherPossibleConditions
        </val:environment>
</cat:property_value>
```

Значение ***PropertyIRDI*** – это действующий IRDI-идентификатор понятия свойства, определенный согласно разделу 9.1.3.2.

Примечание 1 — Присвоение XML-атрибута **property_ref** является обязательным.

Значение ***PropertyIRDIs*** – это список IRDI-идентификаторов понятия свойства согласно разделу 9.1.3.2, который позволяет описывать путь от исходного свойства (определяемого с помощью XML-атрибута **property_ref**), типом данных которого является тип ссылочного класса, до целевого свойства, который участвует в описании вводимого суб-экземпляра класса того экземпляра, для которого исходное свойство определено.

Примечание 2 — Присвоение XML-атрибута **subitem_path_property_ref** является необязательным. Оно может использоваться только для идентифицированного свойства (XML-атрибут **property_ref**), для которого основополагающим типом данных является тип ссылочного класса (см. раздел 8.3.10).

Примечание 3 — XML-атрибут **subitem_path_property_ref** может использоваться как простой способ представления введенной пары «свойство-значение» (см. пример в разделе D.2.1.1 настоящего приложения).

Значения ***DependentValueRepresentation*** и ***ConditionValueRepresentation*** относятся к любому виду типизированного значения, определенному в данном обязательном приложении.

Примечание 4 — Значения DependentValueRepresentation и ConditionValueRepresentation являются обязательными.

Зависимость данного свойства от некоторых других свойств характеризуется с помощью XML-элемента **val:environment,** который содержит по крайней мере один элемент **val:property_value.** Другие условия могут определяться в области значений ***OtherPossibleConditions***, используя тот же самый элемент **val:property_value**.

Примечание 5 — Условие представляется как пара «общее свойство-значение»: ссылка на IRDI-идентификатор свойства и типизированное значение.

Примечание 6 — Условное свойство определено в разделе 6.7.4.

Примечание 7 — Присвоение XML-элементов **cat:property_value, val:property_value** и **val:environment** является обязательным.

Примечание 8 — XML-элементы **val:property_value** и **val:environment** определены в разделе 5.2.4 ИСО/ТС 29002-10:2009.

Пример — Нижеследующее относится к представлению значения зависимого свойства: это свойство идентифицируется с помощью IRDI-идентификатора "0123-ABC#02-LIFETIME#1", а связанное значение – это значение целой меры (см. раздел D.1.17), выражаемое в секундах (элемент UOM_code). Это значение зависит от единственного условия. Условное свойство идентифицируется с помощью IRDI-идентификатора "0123-ABC#02-LOAD#3", и связанным с ним значением является значение действительной меры (см. раздел D.1.16).

```
<cat:property_value property_ref="0123-ABC#02-LIFETIME#1">
        <val:measure_single_number_value UOM_code="s">
                <val:integer_value>1000000</val:integer_value>
        </val:measure_single_number_value>
        <val:environment>
                <val:property_value property_ref="0123-ABC#02-LOAD#3">
                        <val:measure_single_number_value UOM_code="N">
                                <val:real_value>1512.37</val:real_value>
                        </val:measure_single_number_value>
                </val:property_value>
        </val:data_environment>
</cat:property_value>
```

D.2.1.4 Свойство экземпляра

OntoML-представление значения свойства, которое определяется как экземплярное, выражается следующим образом:

```
<cat:property_value property_ref="PropertyIRDI" subitem_path_property_ref="PropertyIRDIs">
        ValueRepresentation
</cat:property_value>
```

Значение ***PropertyIRDI*** – это действующий IRDI-идентификатор понятия свойства, определенный согласно разделу 9.1.3.2.

Примечание 1 — Присвоение XML-атрибута **property_ref** является обязательным.

Значение ***PropertyIRDIs*** – это список IRDI-идентификаторов понятия свойства согласно разделу 9.1.3.2, который позволяет описывать путь от исходного свойства (определяемого с помощью XML-атрибута **property_ref**), типом данных которого является тип ссылочного класса, до целевого свойства, который участвует в описании вводимого суб-экземпляра класса того экземпляра, для которого исходное свойство определено.

Примечание 2 — Присвоение XML-атрибута **subitem_path_property_ref** является необязательным. Оно может использоваться только для идентифицированного свойства (XML-атрибут **property_ref**), для которого основополагающим типом данных является тип ссылочного класса (см. раздел 8.3.10).

Примечание 3 — XML-атрибут **subitem_path_property_ref** может использоваться как простой способ представления встроенных пар значение/свойство экземпляра (см. пример в разделе D.2.1.1 настоящего приложения).

Значение ***ValueRepresentation*** (обязательное) относится к любому виду типизированного значения, определенному в настоящем приложении.

Примечание 4 — Экземплярное свойство определено в разделе 6.7.5.

Примечание 5 — Элемент **cat:property_value** определен в разделе 5.2.4 ИСО/ТС 29002-10:2009.

Пример — Нижеследующее относится к представлению значения экземплярного свойства: это свойство идентифицируется с помощью IRDI-идентификатора "0123-ABC#02-PRICE#1", а связанное значение – это значение действующей валюты (см. раздел D.1.13).

```
<cat:property_value property_ref="0123-ABC#02-PRICE#1">
    <val:currency_value currency_code="EUR">
        <val:real_value>10.53</val:real_value>
    </val:currency_value>
</cat:property_value>
```

D.2.2 Представление экземпляров класса

В данном подразделе определено OntoML-представление:

— класса элементов и экземпляров условного класса элементов;

— класса функциональных моделей и экземпляров производного класса функциональных моделей.

D.2.2.1 Представление класса элементов и экземпляров условного класса элементов

OntoML-представление экземпляров класса элементов или условного класса элементов имеет следующий вид:

```
<cat:item      class_ref="ClassIRDI"
               local_id="LocalId"
               data_specification_ref="DataSpecificationIRDI"
               is_global_id="GloballyIdentifiedBoolean">
    <cat:classification_ref>ClassIRDI</cat:classification_ref>
    ...
    OtherPossibleClassificationReferences
    ...
    <cat:reference      reference_number="StringReferenceNumber"
                        organization_code="StringOrganizationCode"
                        organization_ref="SupplierIRDI">
        <cat:designation>
            <val:local_string>
                <val:content>DesignationString</val:content>
            </val:local_string>
        </cat:designation>
    </cat:reference>
    ...
    PropertyValue(s)Representation
    ...
</cat:item>
```

Примечание 1 — Класс элементов и условный класс элементов определены в разделе 6.7.2.1.

Примечание 2 — Элемент **cat:item** определен в разделе 5.2.3 ИСО/ТС 29002-10:2009.

Элемент ***Class_ref:*** Значение ***ClassIRDI*** и элемент ***classification_ref:*** Значение ***ClassIRDI*** относится к классу элементов или IRDI-идентификатору понятия действующего условного класса элементов, определенных в разделе 9.1.3.1.

Примечание 3 — Присвоение XML-атрибута **class_ref** является обязательным.

Значение ***Local Id*** относится к локальному идентификатору: он определяется для заданного элемента (XML-атрибут **local_id**), ссылка на который дается с помощью свойства (XML-атрибут **item_local_ref**), чьей областью значений является тип ссылочного класса.

Примечание 4 — XML-атрибут **local_id** используется в том случае, если класс предполагается использовать для связи со значением свойства (см. раздел D.1.25), поэтому это присвоение является необязательным.

Значение ***DataSpecificationIRDI*** относится к действующему IRDI-идентификатору описания класса данных, определенному в соответствии с ИСО/ТС 29002-5.

Примечание 5 — Присвоение XML-атрибута **data_specification_ref** является необязательным.

Значение ***GloballyIdentifiedBoolean*** – это возможное булево значение, которое определяет, будет ли элемент идентифицирован глобально или он будет идентифицирован посредством составных компонентов. Когда это булево значение ложно (false), то элемент – это группа компонентов, чей идентификатор должен содержать связанные с ним номера ссылок (см. XML-атрибут **reference_number**), при их наличии, с последующими ссылочными номерами составляющих элементов, рекурсивно рассчитываемых до тех пор, пока эти элементы (для которых существует XML-атрибут **is_global_id)** не станут истинными (true), после чего они будут идентифицированы как собственные. Если они существуют, то связанный ссылочный номер будет содержать достаточно информации для однозначной идентификации этих элементов, будь они компонентами или системой.

Примечание 6 — Присвоение XML-атрибута **global_id** является необязательным.

Элемент **cat:classification_ref** (дополнительный) определяет возможную ссылку (выполняемую с использованием действующего IRDI-идентификатора понятия класса характеристик, определенного в соответствии с разделом 9.1.3.1 настоящего стандарта) для класса категорий, определенного в данном классификаторе. При необходимости нескольких ссылок они могут вводиться с помощью той же структуры, что и для области ***OtherPossibleClassificationReferences.***

Примечание 7 — Класс категорий определен в разделе 6.7.2.1.

XML-элемент **cat:reference** (дополнительный) предназначен для описания ссылочного номера, присваиваемого поставщиком информации.

Примечание 8 — Описание ссылки на элемент с использованием XML-элемента **cat:reference** является необязательным.

Значение ***StringReferenceNumber*** – это строка, присваиваемая поставщиком информации для однозначной идентификации элемента.

Примечание 9 — Если XML-элемент **cat_reference** предоставляется, то XML-атрибут **reference_number** является обязательным.

Значение ***StringOrganizationCode*** – это код, присваиваемый организацией ссылочному номеру. Формат подобных кодов и системы для их присвоения в настоящем стандарте не рассматривается.

Примечание 10 — XML-атрибут присвоения **organization_code** является необязательным.

Значение ***SupplierIRDI*** относится к действующему IRDI-идентификатору описания данных поставщика, определенному в соответствии с разделом 9.1.1. Он соответствует IRDI-идентификатору организации, которая присваивает ссылочный номер.

Примечание 11 — XML-атрибут присвоения **organization_ref** является необязательным.

Значение ***DesignationString*** – это обозначение элемента, которое может переводиться на различные языки (см. раздел D.1.4 для ознакомления локализуемых представлений строки).

Примечание 12 — XML-атрибут присвоения **cat:designation** является необязательным.

Значение ***PropertyValue(s)Representation*** – это место, где определяются представления значения свойства (см. разделы D.2.1.1 - D.2.1.3).

Пример — Далее приводится пример представления простого экземпляра класса, идентифицируемого с помощью IRDI-идентификатора "0123-ABC#01-HEXASCREW#1". Локальный идентификатор ("ITEM_1") также присваивается этому экземпляру класса, который связан с двумя классами (идентифицируемыми с помощью IRDI-идентификаторов "4444-XYZ#01-SCREW#1" и "5555-UVW#01-BOLT#1") и принадлежит двум различным классификационным схемам. Наконец, данный экземпляр класса описывается с использованием двух значений свойства.

```
<cat:item class_ref="0123-ABC#01-HEXASCREW#1" local_id="ITEM_1">
        <cat:classification_ref>4444-XYZ#01-SCREW#1</cat:classification_ref>
        <cat:classification_ref>5555-UVW#01-BOLT#1</cat:classification_ref>
        <cat:property_value property_ref="0123-ABC#02-DIAMETER#1">
                <val:measure_single_number_value UOM_code="mm">
                        <val:integer_value>5</val:integer_value>
                </val:measure_single_number_value>
        </cat:property_value>
        <cat:property_value property_ref="0123-ABC#02-LENGTH#1">
                <val:measure_single_number_value UOM_code="mm">
                        <val:integer_value>20</val:integer_value>
                </val:measure_single_number_value>
        </cat:property_value>
</cat:item>
```

D.2.2.2 Представление функциональной модели и экземпляров производных функциональных моделей

OntoML-представление экземпляров класса функциональных моделей и класса производных функциональных моделей определяется следующим образом:

```
<cat:item       class_ref="ClassIRDI"
                local_id="LocalId"
                data_specification_ref="DataSpecificationIRDI"
                is_model="IsModelBoolean"
                created_view="FunctionalViewIRDI"
                view_of="LocalItemRef">
        <cat:classification_ref>ClassIRDI</cat:classification_ref>
        OtherPossibleClassificationReferences
        ...
        PropertyValue(s)Representation
        ...
</cat:item>
```

Примечание 1 — Класс функциональных моделей определен в разделе 6.7.3.2.

Примечание 2 — Элемент **cat:item** определен в разделе 5.2.3 ИСО/ТС 29002-10:2009.

Элемент **Class_ref:** Значение ***ClassIRDI*** и ***classification_ref:*** Значение ***ClassIRDI*** относится к классу элементов или к IRDI-идентификатору понятия действующего условного класса элементов, определенного в соответствии с разделом 9.1.3.1.

Примечание 3 — XML-атрибут присвоения **class_ref** является обязательным.

Элемент **LocalId** относится к локальному идентификатору, определенному для данного элемента **item** (XML-атрибут **local_id**), ссылка на который дается с помощью свойства (XML-атрибут **item_local_ref**), чья область значений принадлежит типу ссылочного класса.

Примечание 4 — XML-атрибут **local_id** используется в том случае, если класс предполагается использовать для связи со значением свойства (см. раздел D.1.25), поэтому это присвоение является необязательным.

Значение ***DataSpecificationIRDI*** предназначено для идентификатора спецификации данных класса (IRDI), определенного в соответствии с ИСО/ТС 29002-5.

Примечание 5 — Присвоение XML-атрибута **data_specification_ref** является выборочным.

Значение ***IsModelBoolean*** позволяет определить, что элемент является экземпляром класса функциональных моделей (т.е. представлением другого элемента). Это значение должно быть истинным (true).

Примечание 6 — Присвоение XML-атрибута **is_model** является обязательным.

Значение ***FunctionalViewIRDI*** относится к действующему IRDI-идентификатору класса

функциональных представлений, определенному в соответствии с разделом 9.1.3.1 настоящего стандарта; подобное функциональное представление определяет точку зрения, описанную с помощью экземпляра класса функциональных моделей.

Примечание 7 — Присвоение XML-атрибута **created_view** является обязательным.

Значение ***LocalItemRef*** относится к локальному идентификатору, определенному (в случае представления экземпляра производного класса функциональных моделей) с помощью локальной ссылки на элемент, для которого текущий экземпляр производного класса функциональных моделей является представлением.

Примечание 8 — XML-атрибут присвоения **view_of** является обязательным для экземпляров производного класса функциональных моделей. Этот атрибут не используется для экземпляров класса функциональных моделей.

Элемент **cat:classification_ref** (дополнительный) определяет возможную ссылку (производимую с использованием действующего IRDI-идентификатора понятия класса характеристик, определенного в соответствии с разделом 9.1.3.1 настоящего стандарта) для класса категорий, определенного в данном классификаторе. Если необходимо получить несколько ссылок, то они должны вводиться с использованием той же структуры в области ***OthePossibleClassificationReferences***.

Примечание 9 — Класс категорий определен в разделе 6.7.2.1.

Значение ***PropertyValue(s)Representation*** – это место, где определяются представления значения свойства (см. раздел D.2.1.4).

Пример — Далее приведен пример действующего представления простого экземпляра класса функциональных моделей, идентифицируемого с помощью IRDI-идентификатора "0123-ABC#01-HEXASCREWPRICE#1". Этот экземпляр описывает ссылочный элемент (идентифицируемый с помощью локальной ссылки "ITEM_1"), определенный с помощью ссылочного класса функциональных представлений (IRDI-идентификатор "0123-ABC#01-PRICEVIEW#1"). Наконец, этот экземпляр класса описывается с использованием единственного представления значения свойства (IRDI-идентификатор "0123-ABC#02-PRICE#1").

```
<cat:item class_ref="0123-ABC#01-HEXASCREWPRICE#1" is_model="true"
    created_view="0123-ABC#01-PRICEVIEW#1" view_of="ITEM_1">
    <cat:property_value property_ref="0123-ABC#02-PRICE#1">
        <val:currency_value currency_code="EUR">
            <val:real_value>10.53</val:real_value>
        </val:currency_value>
    </cat:property_value>
</cat:item>
```

D.3 Действующее представление расширенных типов данных, определенных в OntoML-языке

В данном разделе определено OntoML-представление значения свойства, чья область значений принадлежит типам расширенных данных согласно ИСО 13584/МЭК 61360.

В OntoML-языке определена онтология расширенных типов данных.

Примечание 1 — Онтология расширенных типов данных определена в приложении Е.

В данной онтологии структура данных каждого расширенного OntoML-типа данных определяется с помощью онтологического понятия класса элементов, с которым связаны описываемые свойства, поэтому представление значения типа расширенных данных определяется с помощью экземпляра класса, который относится к классу элементов (определенному в онтологии), определяющему связанную структуру значений.

Примечание 2 — Представление экземпляров класса элементов (XML-элемент **cat:item**) определено в разделе D.2.2.1 настоящего приложения.

Общая структура ссылки на представление значения типа расширенных данных определяется следующим образом:

```
<val:item_reference_value item_local_ref="LocalIdRef"/>

<cat:item class_ref="ExtendedDataTypeValueStructureClassIRDI" local_id="LocalId">
...
</cat:item>
```

Элемент **LocalId** относится к локальному XML-идентификатору, предназначенному для использования в качестве метки для значения свойства, чья область значений принадлежит типу ссылочного класса. Ссылка приводится с помощью присвоения значения ***LocalIdRef*** XML-атрибуту **item_local_ref**.

Примечание 3 — Элемент LocalId/LocalIdRef аналогичен ссылочному XML механизму ID/IDREF.

Значение ***LocalId*** определено согласно разделу 3.3.8 Части 2: «Типы данных» в XML-схеме.

Значение ***LocalIdRef*** определено согласно разделу 3.3.6 Части 2: «Типы данных» в XML-схеме.

Примечание 4 — Значение для типа ссылочного класса (XML-элемент **val:item_reference_value**) определено в разделе D.1.25 настоящего приложения.

Значение ***ExtendedDataTypeValueStructureClassIRDI*** – это IRDI-идентификатор класса структуры значений расширенных данных (действующий идентификатор понятия класса, определенный в соответствии с приложением E настоящего стандарта).

В последующих подразделах для каждого типа доступных данных определяется соответствующая структура представления значений (экземпляров).

D.3.1 Представления значений для типов расширенных данных согласно ИСО 13584/МЭК 61360

В данном подразделе определены представления значений для типов расширенных данных (см. раздел 8.3.13) в соответствии со структурой данных, определенной в OntoML-онтологии типов расширенных значений.

D.3.1.1 Тип данных placement

OntoML-представление значений свойства, чья область значений принадлежит типу данных **placement**, выражается следующим образом:

```
<cat:item      class_ref="0112-1---13584_32_1#01-PLACEMENT#1"
               local_id="LocalId">
        <cat:property_value property_ref="0112-1---13584_32_1#02-PLACEMENT:LOCATION#1">
               <item_reference_value item_local_ref="CART_P"/>
        </cat:property_value>
</cat:item>

<cat:item class_ref="0112-1---13584_32_1#01-CARTESIAN_POINT#1" local_id="CART_P">
        <cat:property_value property_ref="0112-1---13584_32_1#02-CARTESIAN_POINT:COORDINATES#1">
               <val:sequence_value>
                      <val:real_value>0</val:real_value>
                      <val:real_value>0</val:real_value>
               </val:sequence_value>
        </cat:property_value>
</cat:item>
```

Примечание — Указанное выше представление является полным в соответствии с онтологией для структуры значений типов данных. В следующем подразделе будет представляться только общая структура каждого типа расширенных данных.

IRDI-идентификатор класса, определяющий структуру значений типа данных **placement**, должен устанавливаться как "0112-1---13584_32_1#01-PLACEMENT#1", как это определено в онтологии для типов внешних данных (см. приложение E).

D.3.1.2 Тип данных axis 1 placement

OntoML-представление значений свойства, чья область значений принадлежит типу данных **axis 1 placement**, выражается следующим образом:

```
<cat:item      class_ref="0112-1---13584_32_1#01-AXIS1_PLACEMENT#1"
               local_id="LocalId">
        ...
        PropertyValue(s)Representation
        ...
</cat:item>
```

IRDI-идентификатор класса, определяющий структуру значений типа данных **axis 1 placement**,

должен устанавливаться как "0112-1--- 13584_32_1#01-AXIS1_PLACEMENT#1", как это определено в онтологии для типов внешних данных (см. приложение Е).

Значение ***PropertyValue(s)Representation*** – это представление свойств класса элементов, который представляет расширенный тип данных ***axis1 placement type*** в соответствии с приложением Е. Эти свойства представляются в разделе D.1.

D.3.1.3 Тип данных axis 2 placement 2D

OntoML-представление значений свойства, чья область значений принадлежит типу данных **axis 2 placement 2D,** выражается следующим образом:

```
<cat:item      class_ref="0112-1---13584_32_1#01-AXIS2_PLACEMENT_2D#1"
               local_id="LocalId">
    ...
    PropertyValue(s)Representation
    ...
</cat:item>
```

IRDI-идентификатор класса, определяющий структуру значений типа данных **axis 2 placement 2D**, должен устанавливаться как **"0112-1---13584_32_1#01-AXIS2_PLACEMENT_2D#1"**, как это определено в онтологии для типов внешних данных (см. приложение Е).

Значение ***PropertyValue(s)Representation*** - это представление свойства класса элементов, который представляет расширенный тип данных ***axis2 placement 2D type*** в соответствии с приложением Е.

D.3.1.4 Тип данных axis 2 placement 3D

OntoML-представление значений свойства, чья область значений принадлежит типу данных **axis 2 placement 3D,** выражается следующим образом:

```
<cat:item      class_ref="0112-1---13584_32_1#01-AXIS2_PLACEMENT_3D#1"
               local_id="LocalId">
    ...
    PropertyValue(s)Representation
    ...
</cat:item>
```

IRDI-идентификатор класса, определяющий структуру значений типа данных **axis 2 placement 3D**, должен устанавливаться как "0112-1---13584_32_1#01-AXIS2_PLACEMENT_3D#1", как это определено в онтологии для типов внешних данных (см. приложение Е).

Значение ***PropertyValue(s)Representation*** – это представление свойства класса элементов, который представляет расширенный тип данных ***axis2 placement 3D*** в соответствии с приложением Е.

D.3.2 Ссылка на внешнее представление элементов согласно ИСО 13584/МЭК 61360

В данном подразделе определено представление ссылки на внешние типы представлений, определенные в онтологии для типов расширенных данных.

D.3.2.1 Тип данных representation reference

OntoML-представление значений свойства, чья область значений принадлежит типу данных **representation reference,** выражается следующим образом:

```
<cat:item      class_ref="0112-1---13584_32_1#01-REPRESENTATION_REFERENCE#1"
               local_id="LocalId">
    ...
    PropertyValue(s)Representation
    ...
</cat:item>
```

IRDI-идентификатор класса, определяющий структуру значений типа данных **representation reference**, должен устанавливаться как "0112-1---13584_32_1#01-REPRESENTATION_REFERENCE#1", как это определено в онтологии для типов внешних данных (см. приложение Е).

Значение ***PropertyValue(s)Representation*** – это представление свойства класса элементов, который представляет расширенный тип данных ***representation reference*** в соответствии с приложением Е.

D.3.2.2 Тип данных program reference

OntoML-представление значений свойства, чья область значений принадлежит типу данных **program reference,** выражается следующим образом:

```
<cat:item      class_ref="0112-1---13584_32_1#01-PROGRAM_REFERENCE#1"
               local_id="LocalId">
   ...
   PropertyValue(s)Representation
   ...
</cat:item>
```

IRDI-идентификатор класса, определяющий структуру значений типа данных **program reference**, должен устанавливаться как "0112-1---13584_32_1#01-PROGRAM_REFERENCE#1", как это определено в онтологии для типов внешних данных (см. приложение E).

Значение ***PropertyValue(s)Representation*** – это представление свойства класса элементов, который представляет расширенный тип данных ***program reference type*** в соответствии с приложением E (раздел D.1).

Приложение E

(справочное)

Онтологическое описание расширенных значений в OntoML-языке

OntoML-модель содержит некоторые ориентированные на применение типы данных, чьи значения не могут непосредственно представляться с помощью элементов, определенных в ИСО/ТС 29002-10. Эти типы данных называются «расширенными типами данных» (см. раздел 8.3.13).

Для представления этих данных в настоящем приложении определена онтология, моделирующая эти данные, поэтому значения для расширенных типов данных будут представляться как экземпляры этих классов элементов.

В настоящем приложении определена структура указанной онтологии вместе с IRDI-идентификаторами, которые будут позволять связывать элементы этих классов с их свойствами, а в приложении D.3 указывается способ представления этих классов элементов с помощью схем ИСО/ТС 29002-10.

Онтология OntoML-языка для расширенных значений в типах данных распадается на два вида классов, а именно:

— классы первого уровня, которые соответствуют фактическим расширенным значениям в OntoML-типах данных;

— классы второго уровня, которые позволяют описывать классы первого уровня точным и однозначным образом.

Пример — Класс axis 1 placement определяется как класс первого уровня и поэтому его можно использовать для определения характеристик области значений свойства, определенного в онтологии продукции, чьим типом данных является axis 1 placement type, который сам по себе описывается с использованием различных свойств, например, связанных со свойством reference axis. Это свойство основополагающего типа данных является сложной структурой (классом), указывающим на свойство направления оси direction. Этот класс рассматривается как класс второго уровня, поскольку он используется в контексте описания класса первого уровня axis 1 placement.

Эта онтология расширенных значений OntoML-типов данных распадается на следующие классы первого уровня:

— класс STEP (пространственного позиционирования);

— класс PLIB (внешнего представления).

В следующем разделе описана общая структура онтологии расширенных значений OntoML-типов.

E.1 Структура онтологии расширенных значений

Модель планирования онтологии расширенных значений OntoML-типов данных представлена на UML-диаграмме (см. рисунок E.1).

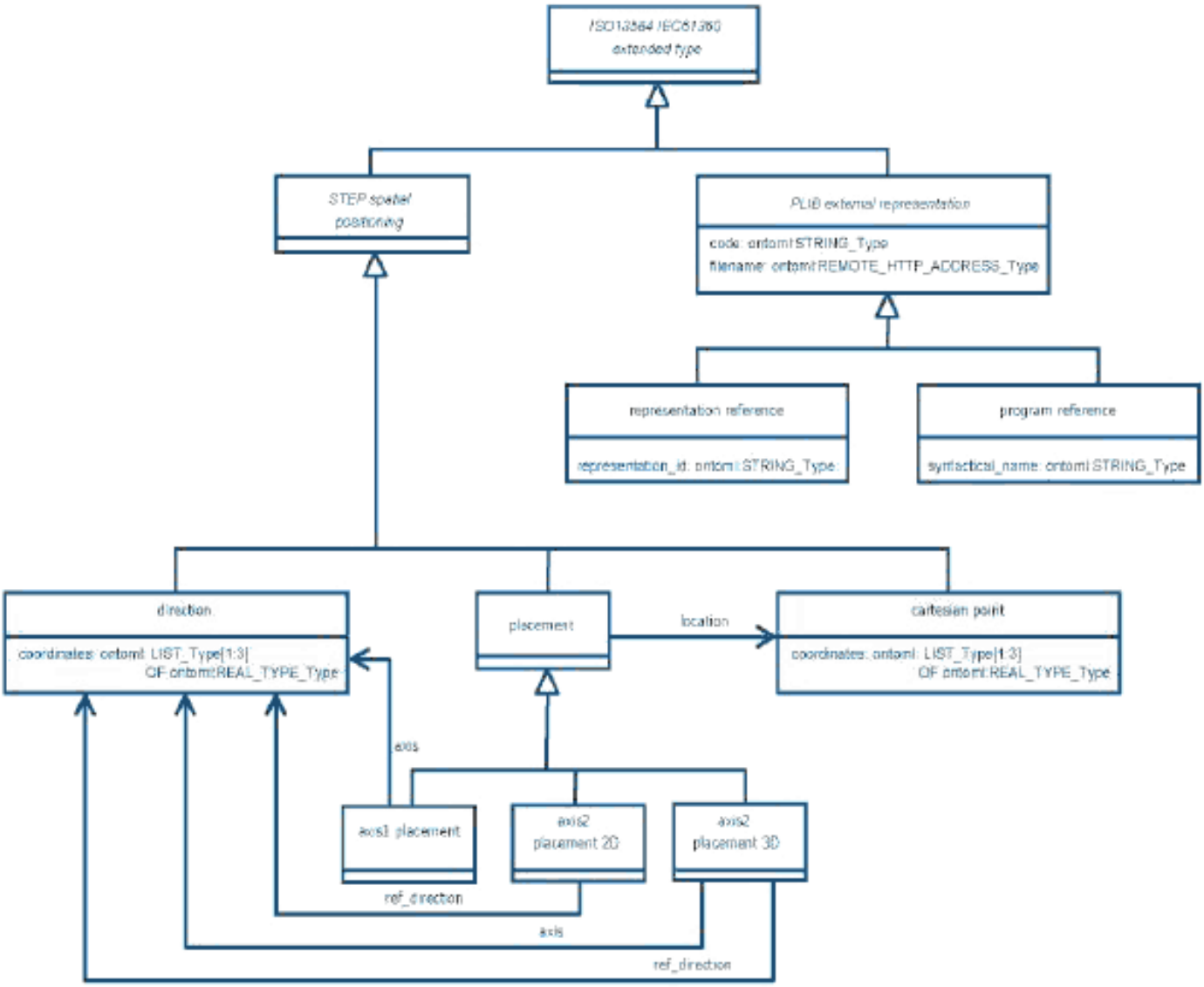


Рисунок Е.1. — Модель планирования онтологии расширенных значений OntoML-типов данных

В этой модели планирования используются следующие обозначения:

— прямоугольниками обозначаются онтологические классы;

— белыми прямоугольниками обозначаются онтологические классы первого уровня, т.е. фактически расширенные типы данных, предназначенные для получения ссылок из свойств, содержащихся в онтологических определениях продукции;

— классы, выделенные курсивом, обозначают представляемые классы;

— выделенные серым фоном прямоугольники обозначают онтологические классы второго уровня, т.е. классы, которые предназначены для получения ссылок от отнологических классов первого уровня или некоторых других онтологических классов второго уровня;

— треугольники обозначают семантические экземплярные (is-a) отношения между онтологическими классами;

— стрелки обозначают связи между онтологическими классами, а соответствующие метки носят имя свойства, которое представляет данную связь;

— свойства простого типа обозначают тип, использующий основной тип OntoML-системы, в котором префикс ***ontoml*** обозначает унифицированный идентификатор ресурса (URI) OntoML-схемы.

Классы первого уровня могут разделяться на следующие категории:

— категория **STEP spatial positioning**:

— категория **axis 1 placement**;

- категория **axis2 placement 2D**;
- категория **axis2 placement 3D**;
- категория **placement**;
- категория **PLIB external representation**;
- категория **program reference**;
- категория **representation reference**.

E.2 Определение расширенных OntoML-значений

В данном разделе определена структура различных расширенных OntoML-значений, определенных в онтологии расширенных OntoML-значений.

E.2.1 Классы первого уровня: структуры расширенных данных

В данном разделе приведена структура классов первого уровня, определенная в онтологии расширенных OntoML-значений.

E.2.1.1 Класс OntoML extended value

Класс (категория) **OntoML extended value** является корневым классом онтологии расширенных значений.

Примечание — Данный класс не является представляемым.

Класс (категория) **OntoML extended value** идентифицируется с помощью следующего IRDI-идентификатора: "0112-1---13584_32_1#01-ONTOML_EXTENDED_VALUE#1".

E.2.1.2 Класс STEP spatial positioning

Класс (категория) **STEP spatial positioning** предназначен для факторизации различных компонентов позиционирования STEP (см. следующие разделы).

Примечание — Данный класс не является представляемым.

Класс (категория) **STEP spatial positioning** идентифицируется с помощью следующего IRDI-идентификатора: "0112-1---13584_32_1 #01-STEP_SPATIAL_POSITIONING#1".

E.2.1.3 Класс placement

Класс (категория) **placement** определяет положение объекта по отношению к координатной системе геометрического контекста.

Примечание — Класс **placement** представляет собой объект ***placement***, определяемый в ИСО 10303-42 (в OntoML-языке).

Класс (категория) **placement** идентифицируется с помощью следующего IRDI-идентификатора: "0112-1---13584_32_1 #01 -PLACEMENT#1".

Определяемое свойство:

Свойство (категория) **placement location** (IRDI: "0112-1---13584_32_1#02-PLACEMENT:LOCATION#1"): Определяет геометрическое положение эталонной точки, например, центра окружности на элементе.

E.2.1.4 Класс axis 1 placement

Класс (категория) **axis 1 placement** определяет направление и местоположение одиночной оси в трехмерной области.

Примечание — Класс **axis 1 placement** представляет объект ***axis1_placement*** из ИСО 10303-42 (в OntoML-языке).

Класс (категория) **axis 1 placement** идентифицируется с помощью следующего IRDI-идентификатора: "0112-1---13584_32_1 #01-AXIS1_PLACEMENT#1".

Определяемое свойство:

Свойство **axis** (IRDI: "0112-1---13584_32_1#02-AXIS1_PLACEMENT:AXIS#1"): Определяет направление локальной оси Z (см. раздел E.2.2.2).

E.2.1.5 Класс axis 2 placement 2D

Класс (категория) **axis 2 placement 2D** определяет местоположение и ориентацию двух взаимно перпендикулярных осей в двумерной области.

Примечание — Класс **axis 2 placement 2D** представляет объект ***axis2_placement_2D*** из ИСО 10303-42 (в OntoML-языке).

Класс (категория) **axis 2 placement 2D** идентифицируется с помощью следующего IRDI-идентификатора: "0112-1---13584_32_1 #01-AXIS2_PLACEMENT_2D#1".

Определяемое свойство:

Свойство **ref direction** (IRDI: "0112-1---13584_32_1#02-AXIS2_PLACEMENT_2D:REF_DIRECTION#1"): Определяет направление локальной оси X (см. раздел E.2.2.2).

E.2.1.6 Класс axis 2 placement 3D

Класс (категория) **axis 2 placement 3D** определяет местоположение и ориентацию двух взаимно перпендикулярных осей в трехмерной области.

Примечание — Класс **axis 2 placement 3D** представляет объект ***axis2_placement_3D*** из ИСО 10303-42 (в OntoML-языке).

Класс (категория) **axis 2 placement 3D** идентифицируется с помощью следующего IRDI-идентификатора: "0112-1---13584_32_1 #01-AXIS2_PLACEMENT_3D#1".

Определяемое свойство:

Класс (категория) **axis** (IRDI: "0112-1---13584_32_1#02-AXIS2_PLACEMENT_3D:AXIS#1"): Определяет направление локальной оси Z (см. раздел E.2.2.2).

Класс (категория) **ref direction** (IRDI: "0112-1---13584_32_1#02-AXIS2_PLACEMENT_3D:REF_DIRECTION#1"): определяет направление локальной оси X (см. раздел E.2.2.2).

E.2.1.7 Класс PLIB external representation

Класс (категория) **PLIB external representation** предназначен для факторизации различных компонентов PLIB-представления (см. следующие разделы).

Примечание 1 — Данный класс не является представляемым.

Класс (категория) **PLIB external representation** идентифицируется с помощью следующего IRDI-идентификатора: "0112-1---13584_32_1#01-PLIB_EXTERNAL_REPRESENTATION#1".

Определяемое свойство:

Класс (категория) **code** (IRDI: "0112-1---13584_32_1#02-PLIB_EXTERNAL_REP:CODE#1"): Идентифицирует внешнее представление в данном классе.

Класс (категория) **file name** (IRDI: "0112-1---13584_32_1#02-PLIB_EXTERNAL_REP:FILE_NAME#1"): Идентифицируется с помощью URI-идентификатора файла, который содержит внешнее представление.

Примечание 2 — Подробнее об этом см. ИСО 13584-24.

E.2.1.8 Класс representation reference

Класс (категория) **representation reference** определяет вид представления, для которого элемент предоставляется в виде внешнего файла, а также имени этого файла, содержащего это представление.

Примечание 1 — В типах данных **DICTIONARY_IN_STANDARD_FORMAT_Type** и **LIBRARY_IN_STANDARD_FORMAT_Type** имеются только протоколы для внешних файлов, которые допускаются либо интегрированной информационной моделью словаря, либо интегрированной информационной моделью библиотеки, индицируемыми с помощью XML-элемента **ontoml_structure** или протоколов обмена представлениями, ссылки на которые даются в XML-элементе **supported_vep**, что определено комплексным XML-типом данных **HEADER_Type**.

Пример — В ИСО 13584-102 определена категория представлений, которая использует обобщенные понятия для описания представления продукции в прикладных протоколах (см. ИСО 10303). Подобное представление предназначено для ссылок на него с помощью экземпляра класса representation reference.

Класс (категория) **representation reference** идентифицируется с помощью следующего IRDI-идентификатора: "0112-1---13584_32_1 #01-REPRESENTATION_REFERENCE#1".

Определяемое свойство:

Свойство **representation id** (IRDI: "0112-1---13584_32_1 # 02-REP_REF:REPRESENTATION_ID#1"): Определяет метку, которая соответствует ссылочному представлению.

Примечание 2 — Подробнее об этом см. ИСО 13584-24.

E.2.1.9 Класс program reference

Класс **program reference** определяет ссылку на алгоритм, предназначенный для представления элемента.

Примечание 1 — Подробнее об этом см. ИСО 13584-24.

Класс (категория) **program reference** идентифицируется с помощью следующего IRDI-идентификатора: "0112-1---13584_32_1 #01-PROGRAM_REFERENCE#1".

Определяемое свойство:

Свойство **syntactical name** (IRDI: "0112-1---13584_32_1 # 02-PROG_REF:SYNTACTICAL_NAME#1"): Определяет имя, с помощью которой должна инициализироваться программа.

Примечание 2 — Подробнее об этом см. ИСО 13584-24.

E.2.2 Классы второго уровня расширенных OntoML-значений

В данном разделе приведена структура классов второго уровня, которая используется для определения различных типов расширенных данных.

E.2.2.1 Класс Cartesian point

Класс **cartesian point** определяет координаты точки в прямоугольной декартовой системе координат или в параметрической области. Координаты точки определяются в одномерной, двухмерной и трехмерной областях (с помощью значений координат в списке).

Примечание 1 — Подробнее об этом см. ИСО 13584-24.

Класс **cartesian point** идентифицируется с помощью следующего IRDI-идентификатора: "0112-1---13584_32_1 #01-CARTESIAN_POINT#1".

Определяемое свойство:

Свойство **coordinates** (IRDI: "0112-1---13584_32_1#02-CARTESIAN_POINT:COORDINATES#1"): Определяет список (из) от 1 до 3 действительных значений, которые задают координаты в прямоугольной декартовой системе координат или в параметрической области.

Примечание 2 — Подробнее об этом см. ИСО 13584-24.

E.2.2.2 Класс Direction

Класс **direction** определяет общее направление вектора в двухмерной и трехмерной областях.

Примечание 1 — Подробнее об этом см. ИСО 13584-24.

Класс **direction** идентифицируется с помощью следующего IRDI-идентификатора: "0112-1---13584_32_1 #01-DIRECTION#1".

Определяемое свойство:

Свойство **direction ratios** (IRDI: "0112-1---13584_32_1#02-DIRECTION:DIRECTION_RATIOS#1"): Определяет список (из) от 2 до 3 действительных значений для общего направления вектора в двухмерной и трехмерной области. Фактические величины компонентов не должны влиять на определяемое направление, значимыми являются только координаты x:y:z или x:y.

Примечание 2 — Подробнее об этом см. ИСО 13584-24.

E.3 Синтез идентификаторов (IRDI) расширенных OntoML-значений

В таблице E.1 приведен перечень IRDI-идентификаторов, определенных для классов, которые принадлежат онтологии OntoML-расширенных значений.

Таблица E.1 — Расширенные OntoML-значения: идентификаторы класса

Наименование OntoML-классов	Идентификатор OntoML-класса (IRDI)
axis 1 placement	0112-1---13584_32_1 #01 -AXIS1_PLACEMENT#1
axis 2 placement 2D	0112-1---13584_32_1 #01 -AXIS2_PLACEMENT_2D#1
axis 2 placement 3D	0112-1---13584_32_1 #01 -AXIS2_PLACEMENT_3D#1
OntoML extended value (abstract)	0112-1---13584_32_1 #01 -ONTOML_EXTENDED_VALUE#1
placement	0112-1---13584_32_1 #01 -PLACEMENT#1
PLIB external representation (abstract)	0112-1---13584 32 1#01-PLIB_EXTERNAL_REPRESENTATION#1

program reference	0112-1---13584_32_1 #01 -PROGRAM_REFERENCE#1
representation reference	0112-1---13584_32_1 #01 -REPRESENTATION_REFERENCE#1
STEP spatial positioning (abstract)	0112-1---13584_32_1 #01 -STEP_SPATIAL_POSITIONING#1

В таблице Е.2 перечислены IRDI-идентификаторы для определения свойств, принадлежащих онтологии расширенных OntoML-значений.

Таблица Е.2 — Расширенные OntoML-значения: идентификаторы свойств

Наименование OntoML-классов	Идентификатор OntoML-класса (IRDI)
axis 1 placement: axis	0112-1---13584_32_1#02-AXIS1_PLACEMENT:AXIS#1
axis 2 placement 2D: ref direction	0112-1---13584 32 1#02-AXIS2_PLACEMENT_2D:REF_DIRECTION#1
axis 2 placement 3D: axis	0112-1---13584_32_1#02-AXIS2_PLACEMENT_3D:AXIS#1
axis 2 placement 3D: ref direction	0112-1---13584 32 1#02-AXIS2_PLACEMENT_3D:REF_DIRECTION#1
cartesian point: coordinates	0112-1---13584 32 1#02-CARTESIAN_POINT_COORDINATES#1
direction: direction ratio	0112-1---13584_32_1#02-DIRECTION:DIRECTION_RATIO#1
placement: location	0112-1---13584_32_1#02-PLACEMENT:LOCATION#1
PLIB external representation: code	0112-1---13584_32_1#02-PLIB_EXTERNAL_REP:CODE#1
PLIB external representation: file name	0112-1---13584 32 1#02-PLIB_EXTERNAL_REP:FILE_NAME#1
program reference: syntactical name	0112-1---13584 32 1#02-PROG_REF:SYNTACTICAL_NAME#1
representation reference: representation id	0112-1---13584_32_1#02-REP_REF:REPRESENTATION_ID#1

Е.4 Формальная модель онтологии расширенных OntoML-значений

Данная формальная модель онтологии расширенных OntoML-значений может быть загружена при задании следующего унифицированного указателя ресурсов (URL): http://www.tc184-sc4.org/implementation information/13584/00032/

Приложение F

(справочное)

Преобразование структуры CIIM-модели из XML-структуры OntoML-языка на язык EXPRESS

Настоящее приложение определяет, как компоненты на OntoML-языке должны преобразовывать данные на EXPRESS-язык. Подобное преобразование позволяет создавать XML-средства для формирования EXPRESS-представлений вариантов документов на OntoML-языке с целью проверки семантической непротиворечивости их содержания по отношению к ограничениям целостности, указанным в CIIM-модели.

F.1 Различия между OntoML- и CIIM- информационными элементами

OntoML-язык является XML-структурой, позволяющей представлять (в рамках используемого варианта XML-документа) все информационные элементы, содержащиеся в физическом файле CIIM-модели, использующей язык EXPRESS.

Как правило, все информационные элементы физического CIIM EXPRESS-файла точно представляются в варианте OntoML-документа, а все ограничения, которые по предположению выполняются для этих информационных элементов, также должны оставаться действующими и для содержания OntoML-документа.

Тем не менее, при разработке OntoML-документа были выявлены четыре различия:

1. Совместное использование и дублирования (sharing Vs duplication) некоторых информационных элементов

Физические OntoML- и CIIM EXPRESS-файлы, онтологические CIIM- понятия определяются только один раз, а упоминаться могут несколько раз. В отношении других частей информации в CIIM-модели заметим, что:

— на EXPRESS-языке некоторые онтологические CIIM- понятия могут использоваться совместно с некоторыми EXPRESS-элементами.

Пример 1 — Элемент item_names может совместно использоваться в некоторых онтологических понятиях.

Примечание 1 — Элемент **item_names** рассмотрен в разделе F.3.9.2.7 ИСО 13584-42:2010.

— в XML-языке было принято решение о том, что каждое онтологическое CIIM-понятие будет только ссылкой на другие понятия этой же онтологии. Все другие части информации, на которые ссылаются онтологические CIIM-понятия в физическом файле CIIM EXPRESS, вводятся в XML-язык, поэтому их содержание, возможно, будет дублированным, если на одну и ту же часть информации будут ссылаться несколько понятий CIIM-онтологии.

Примечание 2 — Дублирование фрагментов информации не изменяет семантики основополагающей CIIM EXPRESS- модели данных.

2. Использование ранее существовавших возможностей XML-языка для определения Интернет-ресурсов

В CIIM-модели для предоставления информации из внешних файлов и способа их обработки были определены некоторые мощные, но сложные методы с использованием EXPRESS-языка. В OntoML-языке подобное представление заменяется использованием метода протоколирования MIME, который вполне достаточен для интерпретации содержимого внешних файлов.

3. Снятие некоторых ограничений, которые не могут проверяться в XML-языке

В CIIM-модели ограничивающее условие **prefix_ordered_class_list** устанавливает, что содержащиеся в экземпляре OntoML-документа классы будут сортироваться таким образом, чтобы не допускалась прямая ссылка одного класса на любой другой класс. Благодаря тому факту, что подобное ограничивающее условие в XML-языке проверяться не может, оно удаляется из OntoML-спецификации и экземпляров OntoML-документов как не отвечающее указанному ограничению. При необходимости этот порядок может компилироваться и гарантироваться, если OntoML-содержание переводится на EXPRESS-язык с целью контроля выполнения ограничивающего условия.

4. Упрощенная CIIM-модель

Предусмотрены следующие упрощения модели:

— предположение относительно обозримости документа;

— упрощение представления преобразования;

— возможность упрощения связи представления документации с любым понятием CIIM-

онтологии;

— удаление CIIM-структур, которые не относятся к OntoML-контексту;

Пример 2 — Типом CIIM-данных в системе OntoML-типа пренебрегают.

— упрощение представления глобальных CIIM-идентификаторов (называемых «базовыми семантическими единицами») в понятиях CIIM-онтологии.

F.2 Иллюстративный пример

В настоящем приложении с помощью OntoML-языка определяется формальное преобразование частей XML-информации, входящей в экземпляр OntoML-документа, в те фрагменты EXPRESS-информации, которые включаются в физический CIIM EXPRESS-файл с целью обмена аналогичной информацией.

В данном разделе на очень простом примере показаны различные компоненты, используемые при подобном преобразовании. Рассмотрим следующую иллюстративную информационную UML-модель (см. рисунок F.1):

Примечание 1 — Подобная информационная модель может представляться на EXPRESS-языке.

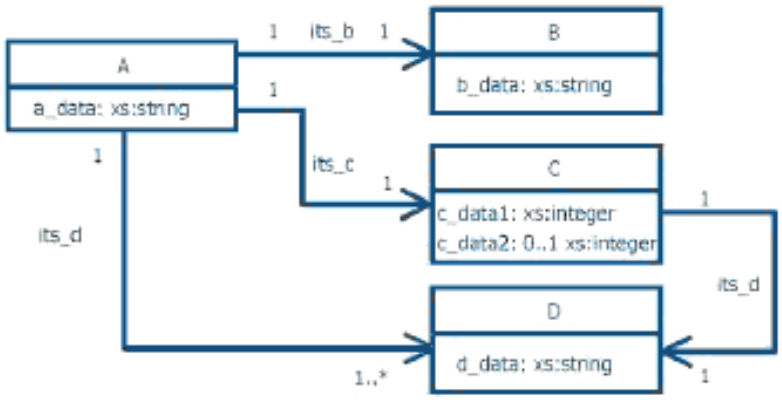


Рисунок F.1 — Пример информационной UML-модели

Комплексный класс *A* определяется с помощью атрибута строки ***a_data***, а также атрибутов ***its_b***, ***its_c*** и ***its_d***, чьи типы данных принадлежат соответственно классам B, C и D. Класс *B* описывается с помощью атрибута строки ***b_data***. Класс C описывается с помощью дополнительных целочисленных атрибутов ***c_data1*** и ***c_data2***, а также атрибутов ***its_d***, чей тип данных принадлежит классу D. Класс D описывается с помощью атрибута строки ***d_data***.

Аналогичное OntoML-представление данной информационной модели, использующей аналогичные принципы преобразования, что и для получения OntoML из CIIM EXPRESS-модели, может быть проиллюстрировано нижеприведенной UML-подобной моделью (см. рисунок F.2).

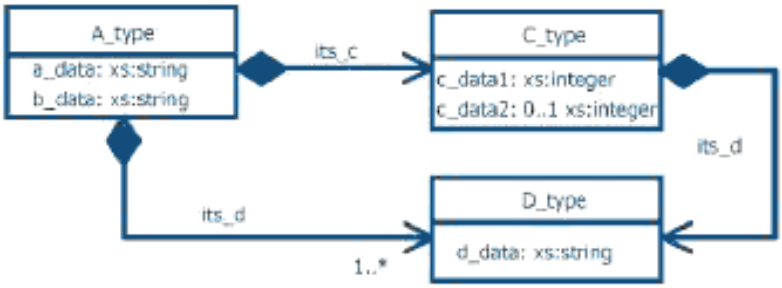


Рисунок F.2 — UML-подобное представление информационной модели

Примечание 2 — Обозначения, используемые в данной UML-подобной модели, аналогичны приведенным в разделе 6.3.1.

Соответствующее представление XML-структуры имеет следующий вид: (см. рисунок F.3):

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <xs:element name="a" type="A_type"/>
    <xs:complexType name="A_type">
        <xs:sequence>
            <xs:element name="a_data" type="xs:string"/>
            <xs:element name="b_data" type="xs:string"/>
            <xs:element name="its_c" type="C_type"/>
            <xs:element name="its_d" type="its_D_type">
            </xs:element>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="C_type">
        <xs:sequence>
            <xs:element name="c_data1" type="xs:int"/>
            <xs:element name="c_data2" type="xs:int" minOccurs="0"/>
            <xs:element name="its_d" type="D_type">
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="its_D_type">
        <xs:sequence>
            <xs:element name="d" type="D_type" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="D_type">
        <xs:sequence>
            <xs:element name="d_data" type="xs:string"/>
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

Рисунок F.3 — Пример XML-структуры

Указанные XML-элементы определены для того, чтобы они играли роль хранилища данных и исходного элемента XML-документа: их типом является комплексный XML-тип ***A_type*** .

Таким образом, имеется возможность представлять одну и ту же информацию в двух форматах: синтаксис экземпляра ISO 10303 21 EXPRESS на основе UML-модели (см. рисунок F.1) и синтаксис XML-документа, основывающегося на использовании синтаксиса XML-структуры (см. рисунок F.4). В таблице F.1 приведен пример использования этих двух форматов представления информации.

Таблица F.1 — XML-экземпляры и соответствующие им ИСО 10303-21-экземпляры

XML document	ISO 10303-21 instances
<a> <a_data>string 1</a_data> <b_data>string 2</b_data> <its_c> <c_data1>10</c_data1> <c_data2>20</c_data2> <its_d> <d_data>string 3</d_data> </its_d> </its_c> <its_d> <d> <d_data>string 4</d_data> </d> <d> <d_data>string 5</d_data> </d> </its_d> 	#1=A('string 1', #2, #6, (#4, #5)); #2=D('string 2'); #3=D('string 3'); #4=D('string 4'); #5=D('string 5'); #6=C(10, 20, #3);

Надписи в таблице: 1 – Экземпляры в XML-документе; 2 - ИСО 10303-21-экземпляры.

Отметим, что определение правил преобразования каждого фрагмента XML-информации в соответствующий фрагмент EXPRESS-информации требует:

— способности создавать экземпляр EXPRESS-представления хранилища XML-элемента, т.е. XML-элемента, называемого **a**, и для ссылки на него;

— способности (в XML-документе) идентифицировать (прямо или косвенно) любую часть введенной информации в хранилище, называемое **a**, и в EXPRESS-файл, для создания экземпляра и идентификации любой части информации, на который ссылаются (прямо или косвенно) из экземпляра элемента **a**;

— способности выражать то, как каждое конкретное значение, представленное на XML-языке, должно преобразовываться, чтобы быть представленным на EXPRESS-языке.

Преобразование, описанное в настоящем приложении, следующее:

— все преобразования начинаются с введения XML-элемента, которым является либо онтологическое CIIM-понятие или исходный элемент словаря. EXPRESS-отображение этого вводимого элемента называется **SELF**; идентификатор вводимого XML-элемента содержит указание места, где правила преобразования представляются в OntoML-языке (см. раздел F.3) и при необходимости – в нотациях XPath /XSLT;

— идентификатор EXPRESS-элементов, на который ссылается EXPRESS-отображение введенного XML-элемента, использует EXPRESS-синтаксис пути, начиная с элемента **SELF** (см. п. F.4);

— при создании экземпляра и представлении значения используется набор специализированных функций, указанных в разделе F.4.3.4.2.

Кроме того, в разделе F.3 определена общая структура вводимых OntoML-элементов.

F.3 Локализация правил преобразования в OntoML-языке

Каждое определение OntoML-элемента связано с (определенным) правилом преобразования (отображения). Это правило выражается с помощью элемента ***annotation***, указанного в спецификации на XML-структуру. Рисунок F.4 иллюстрирует положение правил преобразования, которое будет определено в комплексном XML-типе ***A_Type*** (см. пример XML-структуры на рисунке F.3).

```
<xs:complexType name="A_type">
    <xs:sequence>
        <xs:element name="a_data" type="xs:string">
            <xs:annotation>
                <xs:appinfo> mapping rule(s)</xs:appinfo>
            </xs:annotation>
        </xs:element>
        <xs:element name="b_data" type="xs:string">
            <xs:annotation>
                <xs:appinfo> mapping rule(s)</xs:appinfo>
            </xs:annotation>
        </xs:element>
        <xs:element name="its_c" type="C_type">
            <xs:annotation>
                <xs:appinfo> mapping rule(s)</xs:appinfo>
            </xs:annotation>
        </xs:element>
        <xs:element name="its_d" type="its_D_type">
            <xs:annotation>
                <xs:appinfo> mapping rule(s)</xs:appinfo>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
</xs:complexType>
```

Рисунок F.4 — Представление преобразования в OntoML-языке

Субэлемент ***appinfo*** элемента ***annotation*** предназначен для сохранения в нем всех правил преобразования, которые имеют два аспекта:

— связь между XML-элементом и соответствующим EXPRESS-атрибутом модели данных ИСО 13584: она определяется с помощью заданных EXPRESS-путей (с использованием метода групповой ссылки и ссылки на атрибуты, рассмотренного в ИСО 10303-11:1994); это является *локализационной частью* преобразования;

— закрепление значения (значений) XML-элемента за соответствующим EXPRESS-элементом в CIIM-модели; это является *численной частью* преобразования.

F.4 Связь между OntoML-элементом и CIIM-атрибутом

Преобразование OntoML в CIIM основано на определении заданных EXPRESS-CIIM-путей и основанных на XPath-языке исходных OntoML-путей. Указанные пути используются для установления соответствия между XML- и EXPRESS-элементами.

F.4.1 Исходный OntoML-путь

Исходный XML-путь определяется положением аннотации. Рисунок F.5 иллюстрирует понятие исходного XML-пути:

```
<xs:element name="b_data" type="xs:string">
    <xs:annotation>
        <xs:appinfo>mapping rule(s)</xs:appinfo>
    </xs:annotation>
</xs:element>
```

Рисунок F.5 — Исходный XML-путь

Аннотация присваевается элементу ***b_data***, а исходный путь – элементу ***B_data***.

F.4.2 Целевой EXRESS-путь

Назначение целевого пути состоит в локализации заданной EXPRESS-структуры, которая

соответствует исходной XML-структуре и определяется положением аннотации. Целевая EXPRESS-структура определяется за счет формирования метода ссылки EXPRESS-атрибута с классическим обозначением в виде точки (".") элемента типа данных для экземпляра сущности, которая должна быть создана.

Примечание — Ссылка на атрибут рассмотрена в разделе 12.7.3 ИСО 10303-11:1994.

OntoML-язык определяет некоторые XML-элементы как глобальные, а другие элементы – как локальные.

Только на глобальные элементы даются ссылки с помощью идентификаторов. Все другие элементы вводятся в глобальные элементы.

Таким образом, в EXPRESS-представлении экземпляра OntoML-документа:

— каждый глобальный OntoML-элемент будет представляться путем создания экземпляра соотвествующего EXPRESS-объекта; этот экземпляр будет обозначать и определять контекст, в котором будет преобразовываться вводимый глобальный XML-элемент;

— каждая часть информации, которая не является глобальным элементом, вводится в глобальный XML-элемент и будет преобразована в EXPRESS-отображение введенного глобального XML-элемента.

Рассмотренная глобальная структура показана на рисунке F.6.

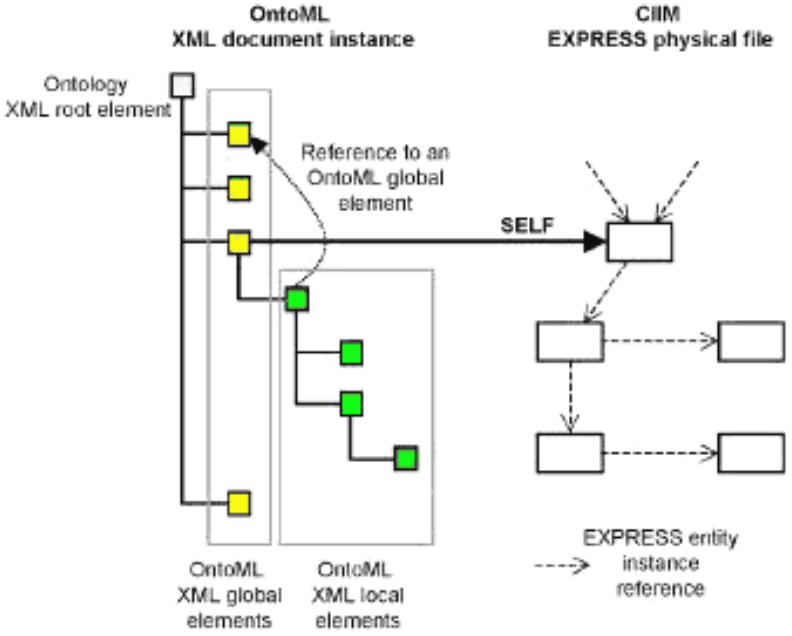


Рисунок F.6 — Глобальные и локальные XML-элементы

Ontology XML root element – Корневой XML-элемент онтологии; OntoML XML document instance – Экземпляр XML-документа на OntoML-языке; OntoML XML global elements – Глобальные XML-элементы на OntoML-языке; Reference to an OntoML global element – Ссылка на глобальный OntoML-элемент; OntoML XML local elements - Локальные XML-элементы на OntoML-языке; Self – EXPRESS-объект; EXPRESS entity instance reference – Ссылка на экземпляр; CIIM EXPRESS physical file - EXPRESS-CIIM – физический файл.

В OntoML-языке определено семь глобальных элементов:

— Элемент **supplier,** представляющий онтологическое понятие поставщика;

— Элемент **class,** представляющий онтологическое понятие класса;

— Элемент **property,** представляющий онтологическое понятие свойства;

— Элемент **data_type,** представляющий онтологическое понятие типа данных;

— Элемент **document,** представляющий онтологическое понятие документа;

— Элемент **ontoml,** представляющий онтологическое понятие корневого элемента и/или библиотеки.

EXPRESS-путь определяется модульным методом: каждый введенный элемент локального элемента содержит локальный путь, который начинается с "." и содержит EXPRESS-путь от EXPRESS-отображения локального XML-элемента до локального элемента EXPRESS-атрибута, соответствующего введенному элементу.

Полный путь от EXPRESS-отображения глобального XML-элемента до EXPRESS-отображения XML-элемента, косвенно введенного в глобальный XML-элемент, формируется путем связывания с глобальным путем, а все локальные пути встречаются при перемещении по древовидной XML-структуре от исходного глобального XML-элемента к конечному введенному XML-элементу.

Структура полного пути формируется из экземпляра понятия CIIM-онтологии (см. раздел F.4.2.1), а также из локальных путей (см. раздел F.4.2.2). Конкретная структура части для введенных элементов, принадлежащих совокупности элементов, рассмотрена в разделе F.4.2.3. Структура полного пути формируется путем объединения этих субпутей согласно разделу F.4.2.4.

F.4.2.1 Экземпляр онтологического CIIM-понятия

В OntoML-языке глобальные XML-элементы представляют собой онтологические CIIM-понятия. Преобразование глобального XML-элемента в соответствующее EXPRESS-отображение осуществляется посредством ключевого слова **SELF**, которое представляет собой экземпляр EXPRESS-объекта и отображение глобального XML-элемента.

В таблице F.2 ниже приведено содержание CIIM SELF-экземпляра в зависимости от OntoML-контекста, в котором он определен. Этот контекст определен двойственно: как XML-комплексный тип и как локальный XML-элемент (см. таблицу F.2).

Таблица F.2 — Содержание элемента (ключевого слова) SELF в его используемом контексте

OntoML-контекст	CIIM EXPRESS-тип элемента "SELF"
Комплексный XML-тип данных: CONTAINED_SUPPLIERS_Type XML-элемент: supplier	Тип supplier_element
Комплексный XML-тип данных: CONTAINED_CLASSES_Type XML-элемент: class	Тип class или один из его подтипов
Комплексный XML-тип данных: CONTAINED_PROPERTIES_Type XML-элемент: property	Тип property_det или один из его подтипов
Комплексный XML-тип данных: CONTAIN ED_DA TA TYPES_Type XML-элемент: datatype	Тип data_type_element
Комплексный XML-тип данных: CONTAINED_DOUMENTS_Type XML-элемент: document XML-корневой элемент: ontoml	Тип document_element Тип dictionary или один из его подтипов

Примечание 1 — Если экземпляр OntoML-документа содержит только спецификацию словаря, то экземпляр **SELF** будет являться экземпляром словаря или CIIM EXPRESS-типом объектных данных **dictionary_in_standard_format**.

Примечание 2 — Если экземпляр OntoML-документа содержит только спецификацию библиотеки, то экземпляр **SELF** будет являться экземпляром библиотеки или CIIM EXPRESS-типом объектных данных **library_in_standard_format**.

Примечание 3 — Если экземпляр OntoML-документа содержит спецификацию и словаря, и библиотеки, то экземпляр **SELF** будет являться экземпляром библиотеки или CIIM EXPRESS-типом объектных данных **library_in_standard_format**.

Пример 1 — В OntoML-языке онтологическое понятие класса представляется с помощью глобального XML-элемента класса. В информационной CIIM EXPRESS-модели это онтологическое

понятие класса представляется с помощью одного экземпляра класса объектных данных или одного из его подтипов. Этот EXPRESS-экземпляр класса называется «SELF–экземпляром» и представляет XML-элемент класса в основанном на EXPRESS-языке множестве.

Глобальные XML-элементы поддерживают полиморфизм, возможно, посредством связанного с ними XML-атрибута **xsi:type**.

Примечание 4 — ***xsi*** означает префикс, связанный со структурной спецификацией XML-диаграммы, которая определяет несколько атрибутов для их непосредственного использования в XML-документе с областью имен <http://www.w3.org/2001/XMLSchema-instance>.

Последнее означает, что EXPRESS-преобразование глобального XML-элемента должно принимать во внимание эту информацию о типе данных, поэтому применимы следующие условия:

— если для определения глобального XML-элемента не используется XML-атрибут **xsi:type**, то SELF-экземпляр будет соответствовать экземпляру EXPRESS-отображения объекта глобального XML-элемента спецификации комплексного типа;

— если для определения глобального XML-элемента используется XML-атрибут **xsi:type**, то SELF-экземпляр будет соответствовать экземпляру EXPRESS-отображения объекта для ссылочного комплексного XML-типа;

Пример 2 — Онтологическое понятие свойства частично может быть определено следующим образом:

```
        </xs:complexContent>
    </xs:complexType>

    <xs:complexType name="NON_DEPENDENT_P_DET_Type">
        <xs:complexContent>
            <xs:extension base="PROPERTY_Type"/>
        </xs:complexContent>
    </xs:complexType>
```

Рассмотрим теперь следующий OntoML-фрагмент:

```
<property xsi:type="NON_DEPENDENT_P_DET_Type" ...>
```

В этом случае экземпляр SELF будет являться экземпляром EXPRESS-отображения, связанного с OntoML-комплексным XML-типом данных NON_DEPENDENT_P_DET_Type, т.е. экземпляром CIIM EXPRESS – типом объектных данных non_dependent_p_det.

F.4.2.2 Информационные элементы онтологии: локальный целевой EXPRESS- путь

Локальный целевой CIIM- путь присваивается каждому локальному XML-элементу, определенному в комплексном XML-типе данных, присваеваемому либо глобальному XML-элементу, либо другому локальному XML-элементу.

Этот путь отображает связь между EXPRESS-отображением локального XML-элемента и EXPRESS-атрибутом.

Примечание 1 — Этот целевой EXPRESS-путь называется «локальным», поскольку он определяет частичное отображение только локально.

Локальный CIIM-целевой путь имеет структуру, иллюстрируемую рисунком F.7:

.local_path.attribute

Рисунок F.7 — Структура локального целевого EXPRESS- пути

где:

— ".": экземпляр EXPRESS-объектного отображения комплексного XML-типа данных, используемый для представления локального XML-элемента;

Примечание 2 — Если комплексный тип данных представляет спецификацию на глобальный XML-элемент, а "." – SELF-экземпляр, то:

— *sub_path:* путь, определенный от объектного EXPRESS-экземпляра до подобного экземпляра, для которого определен целевой атрибут;

— *attribute:* имя EXPRESS-атрибута, для которого глобальный CIIM-путь определяет преобразование (отображение).

Пример — Для представления любой области значений свойства можно использовать реальную меру. В OntoML-языке эта мера представляется на основе комплексного XML-типа данных REAL_MEASURE_TYPE_Type, который определяет локальный XML-элемент unit, представляющий собой

специальную единицу измерений реальной меры. Преобразование единиц XML-элемента в соответствующий EXPRESS-атрибут определяется следующим образом:

.UNIT

Экземпляр "." является экземпляром EXPRESS-отображения, связанного с OntoML-комплексным XML-типом данных REAL_MEASURE_TYPE_Type, т.е. экземпляром CIIM EXPRESS-типов объектных данных real_measure_type.

F.4.2.3 Локальный целевой EXPRESS-путь для индексации группы XML-элементов

Некоторые XML-элементы определяются как группы других введенных XML-элементов, которые соответствуют EXPRESS-атрибутам, тип которых соответствует типу группы. Каждый из этих введенных элементов соответствует одному элементу соответствующей EXPRESS-группы, поэтому необходимо определить отображение между каждым введенным XML-элементом и соответствующим элементом EXPRESS-группы.

Для этих целей EXPRESS-пути выполняются с использованием совокупного индексированного EXPRESS-оператора.

Примечание 1 — Совокупный индексированный оператор определен в разделе 12.6.1 ИСО 10303-11:1994.

Этот оператор состоит из индексированного группового значения (целевой EXPRESS-атрибут) и спецификации, которой должна присваиваться либо целочисленная переменная "i", либо целое постоянное число.

В случае целочисленной переменной диапазон ее значений должен строго определяться следующим образом:

— минимальное значение должно быть равно минимальному граничному значению целевой структуры группы, если она является массивом; в другом случае оно должно быть равно 1;

— максимальное значение должно быть равно минимальному граничному значению (за вычетом 1) и с добавлением числа введенных XML-элементов, которые появляются в XML-элементе, для которого определено преобразование.

Пример — Данный пример иллюстрирует EXPRESS-путь для индексированной группы XML-элементов.

```
<xs:element name="translation" type="TRANSLATION_Type" minOccurs="0">
    <xs:annotation>
        <xs:appinfo>.ADMINISTRATION.TRANSLATION[i]</xs:appinfo>
    </xs:annotation>
</xs:element>
<xs:complexType name="TRANSLATION_Type">
    <xs:sequence>
        <xs:element name="translation_data" type="TRANSLATION_DATA_Type"
            maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="TRANSLATION_DATA_Type" abstract="false">
    <xs:sequence>
        ...
        <xs:element name="translation_revision" type="REVISION_TYPE_Type">
            <xs:annotation>
                <xs:appinfo>.TRANSLATION_REVISION</xs:appinfo>
            </xs:annotation>
        </xs:element>
        ...
    </xs:sequence>
</xs:complexType>
```

OntoML-элемент translation соответствует (.ADMINISTRATION.TRANSLATION[i]) EXPRESS-атрибуту translation, определенному в объектном EXPRESS-типе данных administrative_data, ссылка на который дается с помощью EXPRESS-атрибута administration. EXPRESS-атрибут типа данных translation является группой объектных EXPRESS-экземпляров класса translation_data. Каждый объектный EXPRESS-экземпляр класса описывается с помощью множества атрибутов, среди которых имеется EXPRESS-атрибут translation_revision. В OntoML-языке каждый элемент группы XML-элементов translation представляется с помощью введенного XML-элемента translation_data. XML-элемент

translation_data определяется в соответствии с комплексным XML-типом данных TRANSLATION_DATA_Type, который описывает XML-элемент translation_revision, чье EXPRESS-отображение предварительно описывалось с помощью EXPRESS-атрибута translation_revision (.TRANSLATION_REVISION).

Примерчание 2 — Индексирование 2D- агрегатной структуры будет выражаться следующим образом: *attribute_name[i][j].*

F.4.2.4 Полная структура целевого EXPRESS-пути

Полный целевой EXPRESS-путь определяет EXPRESS-путь от экземпляра класса **SELF** до EXPRESS-атрибута, который является отображением конечного локального XML-элемента.

Полный путь от EXPRESS-отображения экземпляра глобального XML-элемента до EXPRESS-отображения XML-элемента (который косвенно вводится в глобальный XML-элемент) создается с помощью объединения экземпляра **SELF** и всех локальных путей, возникающих при перемещении по древовидной XML-структуре от начального глобального XML-элемента к конечному введенному XML-элементу.

Полный целевой CIIM-путь имеет структуру, показанную на рисунке F.8:

SELF. sub_path. Attribute

Рисунок F.8 — Структура полного целевого EXPRESS-пути

где:

— *SELF:* EXPRESS-экземпляр, соответствующий глобальному OntoML-элементу;

— *sub_path:* Объединение всех локальных путей, встречающихся при перемещении по древовидной XML-структуре, начиная от начального глобального XML-элемента и заканчивая конечным XML-элементом;

— *attribute:* Имя целевого EXPRESS-атрибута, соответствующее конечному введенному XML-элементу.

Пример 1 — Класс является онтологическим CIIM-понятием, который связывается со своим именем. В информационной CIIM EXPRESS-модели это имя представляется в объектном типе данных ITEM_NAMES в виде атрибута preferred_name. В OntoML-языке данное онтологическое CIIM-понятие представляется с помощью глобального XML-элемента класса с моделью содержания, определяемой с помощью комплексного типа данных CLASS_Type, который определяет XML-элемент, называемый preferred_name и представляемый этим именем класса. Полный целевой EXPRESS-путь будет полностью определяться следующим выражением:

SELF. NAMES. PREFERRED_NAME

Этот полный целевой CIIM-путь определяет связь между элементом preferred_name онтологического понятия OntoML-класса и атрибутом preferred_name EXPRESS-объекта item_names. SELF представляет объектный экземпляр класса (или экземпляр одного из его подтипов).

Пример 2 — Предположим, что переведенный локальный XML-элемент, представленный в примере раздела F.4.2.3, непосредственно вводится в глобальный XML-элемент, представляющий, например, онтологическое понятие свойства. Полный путь, который ведет к локальному XML-элементу translation_revision, описывается следующим выражением:

SELF. ADMIN IS TRA TION. TRANSLA TION[i]. TRANSLA TION_REVISION

F.4.3 Присвоение значения OntoML-элемента атрибуту EXPRESS-языка

Присвоение значения OntoML-элемента атрибуту EXPRESS-языка требует определения:

— OntoML-источника и целевого атрибута EXPRESS-пути, определяющих отображающие информационные элементы;

— оператора присвоения;

— синтаксиса для получения доступа к информационным единицам в совместимом с OntoML-языком экземпляром XML-документа;

— специфических EXPRESS-конструкторов для тех информационных элементов, которые непосредственно не представлены в OntoML-языке в соответствии с CIIM-моделью.

В данном разделе рассмотрены все аспекты присвоения.

F.4.3.1 Оператор присвоения

Присвоение значения, представленного в OntoML-документе, целевому EXPRESS-пути производится с помощью оператора присвоения ":=" .

F.4.3.2 Операция присвоения

Присвоение EXPRESS-значения целевому EXPRESS-пути определяется только для того XML-элемента, который определяет конечную цель для соответствующего полного целевого EXPRESS-пути. Это присвоение проводится с использованием следующего синтаксиса:

EXPRESS target path := EXPRESS value

где:

— *EXPRESS target path:* Локальный целевой путь, присваиваемый XML-элементу;

Примечание — Никакое другое отображение не определено для введенного XML-элемента данного XML-элемента.

— *EXPRESS value:* Значение, которое должно быть либо ссылочным (простое значение), либо расчетным (комплексное значение), полученное с помощью специальной отображающей функции.

Пример 1 — Номер редакции онтологического понятия класса представляется с помощью EXPRESS-атрибута, называемого «редакцией», чьим типом данных является строка. Этот номер представляется в OntoML-языке с помощью XML-элемента (редакции), чьей моделью содержания также является простая строка. Отображение между OntoML-представлением редакции состоит в присвоении значения OntoML-строки редакции EXPRESS-атрибуту этой редакции.

Пример 2 — Предпочтительное имя онтологического понятия класса представляется с помощью EXPRESS-атрибута, называемого preferred_name, чьим типом данных является тип translatable_label, представляемый в OntoML-языке с помощью XML-элемента preferred_name, чья модель содержания с целью упрощения не представляется с использованием тех же компонентов. Следовательно, отображение не может представляться просто использованием целевого EXPRESS-пути. Должна использоваться специальная отображающая функция.

F.4.3.3 Выборка OntoML-информации

Определение отображения между экземпляром OntoML-документа и EXPRESS-экземплярами предназначено для выборки XML-данных документа, с последующим их присвоением объектным EXPRESS-атрибутам.

Для этой цели правила отображения OntoML-языка используют синтаксис XPath. Каждый XPath определяется локально для XML-элемента, для которого определено правило отображения. Синтаксис XPath ограничивается следующими компонентами:

— .: возврат текущего узла (вершины);
— ***@attribute_name:*** возврат значения атрибута <*attribute_name*> для текущего узла;
— *: возврат всех дочерних элементов текущего элемента, вне зависимости от их имен;
— /: разделитель, используемый для определения этапов локализации XPath;
— ***element_name***: возврат всех дочерних узлов <*element_name*> контекстно-зависимого узла.

F.4.3.4 Присвоение OntoML-информации целевым EXPRESS-путям

В данном разделе определены различные способы, используемые для присвоения значений, ссылки на которые даются из XML-документа на целевые EXPRESS-пути.

F.4.3.4.1 Присвоение простого OntoML-значения простому EXPRESS-атрибуту

Простое XML-значение для его присвоения EXPRESS-атрибуту безусловно выполняется для тех XML-элементов, чья модель содержания определена как простая. С целью упрощения подобное простое присвоение не будет требовать использования оператора присвоения.

Пример — Дата исходного определения онтологического CIIM-понятия отображается следующим образом:

```
<xs:element name="class" type="CLASS_Type" maxOccurs="unbounded">
    <xs:annotation>
        <xs:appinfo>SELF</xs:appinfo>
    </xs:annotation>
</xs:element>
<xs:complexType name="CLASS_Type" abstract="true">
    <xs:sequence>
    ...
```

```
<xs:element name="date_of_original_definition" type="DATE_TYPE_Type" minOccurs="0">
    <xs:annotation>
<xs:appinfo>.TIME_STAMPS.DATE_OF_ORIGINAL_DEFINITION</xs:appinfo>
    </xs:annotation>
</xs:element>
    ...
</xs:complexType>
```

Рассмотрим теперь следующий OntoML-фрагмент:

```
<class xsi:type="..." id="...">
...
    <revision>1</revision>
...
</class>
```

Отображение значения номера OntoML-редакции на соответствующий полный целевой EXPRESS-путь определяется следующим выражением:

SELF.TIME_STAMPS.DATE_OF_ORIGINAL_DEFINITION := 1

F.4.3.4.2 Присвоение OntoML-значения комплексному EXPRESS-атрибуту

Некоторые информационные элементы в соответствии с CIIM-моделью непосредственно в OntoML-языке не представляются. Преобразование элементов не может выражаться только с помощью EXPRESS-целевых путей и присвоения значений простым XML-элементам, что требует определения отображающих функций с целью извлечения информации из OntoML-документа и ее обработки, а также для присвоения значения экземпляра атрибуту, на который ссылка дается с помощью полного EXPRESS-целевого пути. Соответствующий алгоритм может быть более или менее сложным.

Примечание — Преобразование (отображение) не определяет алгоритм выполнения каждой из отображающих функций, а только способ его записи и характеристики.

Комплексный EXPRESS-атрибут является атрибутом, чей тип данных – это тип объектных данных. Присвоение OntoML-значения подобному атрибуту требует создания совместимого по типу экземпляра. Общая структура подобного присвоения определяется следующей записью:

EXPRESS target path := <function_name>({parameters}),

где:

— ***EXPRESS-target path***: локальный целевой путь, закрепленный за XML-элементом, а типом данных целевого атрибута является EXPRESS-объект;

— ***<function_name>***: отображающая функция, которая служит для создания множества EXPRESS-экземпляров и закрепления одного из них за атрибутом, имеющим ссылку в EXPRESS-целевом пути;

— ***{parameters}***: множество эффективных параметров, соответствующих значению XML-элемента (элемента или атрибута), выбираемого из представления OntoML-документа с помощью оператора XPath.

Пример — Предпочтительное имя онтологического понятия класса представляется с помощью EXPRESS-атрибута, называемого preferred_name, чьим объектным типом данных является элемент translatable_label. Они представляются в OntoML-языке с помощью XML-элемента preferred_name, чья модель для упрощения содержания не представляется с помощью тех же компонентов. Соответственно, преобразование не может представляться просто с помощью EXPRESS-целевого пути. Специальная отображающая функция использует:

```
<xs:element name="class" type="CLASS_Type" maxOccurs="unbounded">
    <xs:annotation>
        <xs:appinfo>SELF</xs:appinfo>
    </xs:annotation>
</xs:element>
<xs:complexType name="CLASS_Type" abstract="true">
    <xs:sequence>
    ...
     <xs:element name="preferred_name" type="PREFERRED_NAME_Type">
         <xs:annotation>
             <xs:appinfo>.NAMES.PREFERRED_NAME :=
                 createLabel(*)</xs:appinfo>
         </xs:annotation>
     </xs:element>
...
</xs:complexType>
```

Рассмотрим теперь его OntoML-фрагмент:

```
<class xsi:type="..." id="...">
...
    <preferred_name>
        <label language="en">bearing</label>
    </preferred_name>
...
</class>
```

Преобразование комплексного значения предпочтительного OntoML-имени в соответствующий полный EXPRESS-целевой путь безусловно определяется следующим выражением:

SELF. NAMES. PREFERRED_NAME := createLabel(*)

Значение EXPRESS-атрибута preferred_name предназначено для использования отображающей функции createLabel, которая берет в качестве эффективного параметра множество вершин (узлов), помечаемых с помощью символа "*" XPath (множества дочерних вершин контекстно-зависимого XML-элемента, т.е. вершин-потомков XML-элемента preferred_name), и процесс (создание EXPRESS-экземпляров) в соответствии с CIIM-моделью. Эта функция выдает значение совместимого по типу EXPRESS-атрибута preferred_name (объектный экземпляр translated_label), который присваивается определенному полному EXPRESS-целевому пути.

В следующем подразделе определены указанные выше отображающие функции.

F.4.3.4.2.1 Базовая семантическая единица (BSU) из преобразования CIIM-идентификатора онтологического понятия

Каждое понятие CIIM-онтологии связано с однозначным идентификатором со структурой, определенной в данной части ИСО 13584.

В OntoML-языке эти идентификаторы представляются в виде строки, тогда как они структурно и описательно определены в CIIM-модели, поэтому мы определяем функцию, которая предназначена для создания ресурсов экземпляров типа объектных данных. Таким образом, мы определяем функцию, которая предназначена для создания указанных выше ресурсов для представления идентификатора понятия CIIM-онтологии из идентификатора, представленного строкой, имеющей следующий вид:

<ontologyConcept>BSUFromId(ontoMLId: string): basic_semantic_unit,

где:

— ***<ontologyConcept>:*** особое понятие CIIM-онтологии, для которой создается CIIM-идентификатор, принимающий следующие значения:

— онтологическое понятие поставщика: "supplier";

— онтологическое понятие класса: "class";

— онтологическое понятие свойства: "property";

— онтологическое понятие типа данных: "datatype";

— понятие документа: "document";

— ***ontoMLId***: строка, представляющая идентификатор OntoML-понятия;

Примечание 1 — Эффективное значение **ontoMLId** предназначено для его извлечения из OntoML-представления документа с использованием локализационного XPath-пути.

— ***basic_semantic_unit***: тип экземпляра, выдаваемый при обращении к функции;

Примечание 2 — Элемент **basic_semantic_unit** определен в разделе F.3.4.2.1 ИСО 13584-42:2010.

В зависимости от выбора понятия <ontologyconcept>, функция **<ontologyConcept>BSUFromId** выдает экземпляр типа:

— объектных данных **supplier_BSU** - для онтологического понятия поставщика;
— объектных данных **class_BSU** - для онтологического понятия класса;
— объектных данных **property_BSU** - для онтологического понятия типа свойства;
— объектных данных **datatype_BSU** - для онтологического понятия типа данных;
— объектных данных **document_BSU** - для онтологического понятия документа.

Если OntoML-идентификатор уже преобразован (отображен) в экземпляре одного из этих CIIM-объектных данных, то он не должен создаваться повторно, однако соответствующий экземпляр CIIM-объекта должен быть извлечен и выдан с помощью соответствующей функции.

Кроме того, в зависимости от идентифицированного понятия CIIM-онтологии имеет место следующее:

— онтологическое понятие класса: если идентифицированный поставщик в идентификаторе OntoML-класса пока еще не преобразован в EXPRESS-экземпляр объекта, то он должен быть создан и затем снабжен ссылкой, в противном случае существующий EXPRESS-экземпляр объекта должен снабжаться только ссылкой;

— онтологическое понятие свойства, типа данных или документа: если идентифицированный поставщик и идентифицированный класс в OntoML-идентификаторе свойства, типа данных или документа пока еще не преобразованы в EXPRESS-экземпляры объектов, то они должны быть созданы и затем снабжены ссылками, в противном случае существующие EXPRESS-экземпляры объектов должны снабжаться только ссылками.

В таблице F.3 приведены OntoML CIIM-идентификаторы онтологических понятий (см. раздел) и соответствующие им CIIM EXPRESS-экземпляры.

Таблица F.3 — Преобразование (отображение) OntoML-идентификаторов

OntoML-идентификаторы	EXPRESS-экземпляры
SupplierId ::= icd oi [opi [opis]] [std]	#supp=SUPPLIER_BSU(CIIMrai, *); *CIIMrai* сформирован из идентификатора ***supplierId*** в соответствии с правилами ИСО 13584-26
classId ::= rai # di #vi	#cl=CLASS_BSU(di, vi, #supp); *#supp* является ссылкой на экземпляр **supplier_BSU**, идентифицированный в части ***rai*** идентификатора ***dictionaryId***
propertyId ::= rai # di # vi	#prop=PROPERTY_BSU(diProp, vi, #cl) ; *diProp* является CIIM-кодом свойства, идентифицированном в части *di* OntoML-идентификатора ***propertyId*** *#cl* является ссылкой на экземпляр **class_BSU** идентифицированный в ***rai*** и части ***di*** OntoML-идентификатора ***propertyId***

Окончание таблицы F.3

OntoML-идентификаторы	EXPRESS-экземпляры
documentId ::= rai # di # vi	#doc=DOCUMENT_BSU(diDoc, vi, #cl); *diDoc* является CIIM-кодом документа, идентифицированном в части ***di*** part of the OntoML- идентификатора ***documentId***. *#cl* является ссылкой на экземпляр **class_BSU** идентифицированный в ***rai*** и части ***di*** OntoML-идентификатора ***propertyId***
datatypeId ::= rai # di # vi	#type=DATA_TYPE_BSU(diType, vi, #cl) ; *diType* является CIIM-кодом типа данных, идентифицированным в части ***di*** OntoML-идентификатора ***datatypeId*** # является ссылкой на экземпляр **class_BSU** идентифицированный в ***rai*** и части ***di*** OntoML-идентификатора ***datatypeId***

ПРИМЕР — Идентификатор онтологического понятия класса представляется в следующем виде:

```
<xs:element name="class" type="CLASS_Type" maxOccurs="unbounded">
        <xs:annotation>
                <xs:appinfo>SELF</xs:appinfo>
        </xs:annotation>
</xs:element>
<xs:complexType name="CLASS_Type" abstract="true">
        ...
        <xs:attribute name="id" type="ClassId" use="required">
                <xs:annotation>
                        <xs:appinfo>.IDENTIFIED_BY := classBSUFromId(string(@id))</xs:appinfo>
                </xs:annotation>
        </xs:attribute>
</xs:complexType>
```

Полным целевым EXPRESS-путем, определенным для XML-атрибута id (идентификатора онтологического понятия класса) и соответствующим ему представлением, таким образом, является:

SELF.IDENTIFIED_BY := classBSUFromId(string(@id))

Последнее означает, что EXPRESS-атрибут identified_by для экземпляра SELF (представляющего онтологическое понятие класса) устанавливается на значение, выдаваемое функцией classBSUFromId, в которой в качестве аргумента используется определенное XPath-значение ("string(@id)"), т.е. значение OntoML-атрибута id .

Рассмотрим следующий OntoML-фрагмент:

```
<class ... id="0002-38491502100024#BEARING#001">
...
</class>
```

Если мы предположим, что объектный экземпляр supplier_BSU уже был создан (#supp), то отображение будет выглядеть следующим образом:

```
#cl = CLASS_BSU('BEARING', '001', #supp);
SELF.IDENTIFIED_BY := #cl
```

где:

— ***#cl: EXPRESS-идентификатор экземпляра class_BSU;***

— ***#supp: EXPRESS-идентификатор экземпляра supplier_BSU (который по предположению существует).***

F.4.3.4.2.2 Класс BSU из отображения класса идентификаторов

CIIM EXPRESS-модель требует получения ссылки на каждый определенный или ссылочный класс от EXPRESS-объекта **dictionary** посредством его атрибута **contained_classes**, для чего используется функция **classBSUsFromIds**, который позволяет извлекать все OntoML-идентификаторы класса и выдавать их соответствующее CIIM-представление (**class_BSU instances**). Соответствующая запись имеет следующий вид:

classBSUsFromIds(ontoMLIds: XPath): LIST OF UNIQUE class_BSU,
где:
— *ontoMLIds:* XPath, позволяющий извлекать все OntoML-идентификаторы класса;
Примечание 1 — В OntoML-языке подобный идентификатор представляется с помощью XML-атрибута.

— *class_BSU:* тип экземпляра, выдаваемый путем запроса этой функции.
Примечание 2 — Элемент **Class_BSU** определен в разделе F.3.6.1.1 ИСО 13584-42:2010.
Примечание 3 — Функция **classBSUsFromIds** используется лишь однократно, в описании отображения XML-элемента **contained_classes**, определенного в комплексном XML-типе данных **DICTIONARY_Type**.

В таблице F.4 приведен иллюстративный пример фунции **classBSUsFromIds** (в предположении, что эта отображающая функция определена в OntoML-контексте XML-элемента **contained_classes**):

.CONTAINED_CLASSES := classBSUsFromIds(*/@id).

Таблица F.4 — OntoML-список представлений идентификаторов класса

OntoML-представление	EXPRESS-экземпляры
<contained_classes> <class id="rai#di1#vi" ...> ... </class> <class id=" rai#di2#vi" ...> ... </class> <class id=" rai#din#vi" ...> ... </class> </contained_classes>	#cl1=CLASS_BSU(di1, vi, #supp); #cl2=CLASS_BSU(di2, vi, #supp); #cln=CLASS_BSU(din, vi, #supp); *#supp* – ссылка на экземпляр supplier_BSU, идентифицированный в *rai частu* идентификатора *dictionaryId* ***classBSUsFromIds*** function result: [#cl1, #cl2, ..., #cln]

F.4.3.4.2.3 Отображение идентификаторов словаря и библиотеки

Функция **dictionaryCodeFromId** позволяет создавать объектный EXPRESS-экземпляр **dictionary_identification** из OntoML-идентификатора. Соответствующая запись имеет следующий вид:

DictionaryCodeFromId(ontoMLDicLibId: string): dictionary_identification

где:

— *ontoMLDicLibId:* XPath, позволяющий получать доступ к OntoML-идентификатору словаря и/или библиотеки;

Примечание 1 — В OntoML-языке подобный идентификатор представляется в виде XML-атрибута.

— *dictionary_identifier:* тип экземпляра, выдаваемый путем запроса этой функции.

Примечание 2 — Элемент **dictionary_identification** определен в разделе 11.5 ИСО 13584-24:2003.

В таблице F.5 приведены идентификаторы словаря и библиотеки (см. раздел 9.1) и их соответствующие CIIM EXPRESS-представления.

Таблица F.5 — Отображение онтологических OntoML-идентификаторов

OntoML-представление	EXPRESS-экземпляры
ontologyId ::= rai # di #vi	#dic=DICTIONARY_IDENTIFICATION(di, vi, revision, #supp); #*supp* – это ссылка на экземпляр **supplier_BSU,** идентифицированный в части *rai* идентификатора ***ontologyId***. Атрибут ***revision*** с помощью данной функции не отображается.

F.4.3.4.2.4 Отображение метки и переведенной метки

Функция **createLabel** позволяет создавать CIIM EXPRESS-ресурсы, соответствующие некоторой незашифрованной информации о метках, возможно, переведенных. Соответствующая запись имеет следующий вид:

createLabel(ontoMLLabel: XPath): translatable_label

где:

— *ontoMLLabel:* XPath, который ссылается на OntoML-множество XML-элементов **label** (возможно, связанного с XML-атрибутом **language**), предназначенных для обработки;

— *translatable_label:* общий тип данных, выдаваемый путем запроса этой функции. В случае непереведенного текста она выдает значение объектного CIIM-экземпляра **label_type,** а в случае переведенного текста - значение CIIM-экземпляра **translated_label**.

Примечание — Элемент **translatable_label** определен в разделе F.4.1.4 ИСО 13584-42:2010.

В таблице F.6 приведены метки и переведенные метки, а также соответствующие им CIIM EXPRESS-экземпляры.

Таблица F.6 — Отображение OntoML-меток и переведенных меток

OntoML-метки	EXPRESS-экземпляры
<…> <label> a label </label> </…>	LABEL('a label') CIIM-представление непереведенной метки является строкой, чьим специфическим типом данных является тип **LABEL.**

<…> <label language="en"> a label </label> <label language="fr"> un label </label> </…>	#tlabel=TRANSLATED_LABEL(('a label', 'un label'), #pt); #pt=PRESENT_TRANSLATIONS((#lc1, #lc2)); #lc1=LANGUAGE_CODE ('en', $); #lc2=LANGUAGE_CODE('fr', $); CIIM-представление переведенной метки состоит из экземпляров следующих трех объектных CIIM EXPRESS-типов данных: **translated_label, present_translations** и **language_code**

Пример — Онтологическое понятие предпочтительного имени класса представляется следующим образом:

```
<xs:element name="class" type="CLASS_Type" maxOccurs="unbounded">
        <xs:annotation>
                <xs:appinfo>SELF</xs:appinfo>
        </xs:annotation>
</xs:element>
<xs:complexType name="CLASS_Type" abstract="true">
        ...
        <xs:element name="preferred_name" type="PREFERRED_NAME_Type">
                <xs:annotation>
                        <xs:appinfo>.NAMES.PREFERRED_NAME := createLabel(*)</xs:appinfo>
                </xs:annotation>
        </xs:element>
        ...
</xs:complexType>
<xs:complexType name="PREFERRED_NAME_Type">
        <xs:sequence>
                <xs:element name="label" type="PREFERRED_NAME_LABEL_Type"
                        maxOccurs="unbounded"/>
        </xs:sequence>
</xs:complexType>
<xs:complexType name="PREFERRED_NAME_LABEL_Type">
        <xs:simpleContent>
                <xs:extension base="PREFERRED_NAME_TYPE_Type">
                        <xs:attribute name="language" type="LANGUAGE_CODE_Type"
                                use="optional"/>
                </xs:extension>
        </xs:simpleContent>
</xs:complexType>
```

Полный целевой EXPRESS-путь, определенный для XML-элемента preferred_name (класса предпочтительных имен), и его соответствующее присвоение имеет следующий вид:

SELF. NAMES. PREFERRED_NAME := createLabel(*)

Рассмотрим теперь его OntoML-фрагмент:

```
<class ...>
...
        <preferred_name>
            <label language="en">roller bearing</label>
        </preferred_name>
...
</class>
```

Результат указанного выше отображения может быть записано как:

```
#tlabel = TRANSLATED_LABEL(('roller bearing'), #pt);
#pt = PRESENT_TRANSLATIONS((#lc));
#lc = LANGUAGE_CODE('en', $);
SELF.NAMES.PREFERRED_NAME := #tlabel
```

F.4.3.4.2.5 Отображение текста и переведенного текста

Функция **create Text** позволяет создавать CIIM EXPRESS-ресурсы, соответствующие некоторой незашифрованной текстовой информации, возможно, переведенной. Соответствующая запись имеет следующий вид:

createText(ontoMLText: XPath): translatable_text

где:

— *ontoMLText:* XPath, который ссылается на OntoML-множество XML-элементов **text** (возможно, связанного с XML-атрибутом **language**), предназначенных для обработки;

— *translatable_Text:* общий тип данных, выдаваемый путем запроса этой функции. В случае непереведенного текста она выдает значение объектного CIIM-экземпляра **text_type,** а в случае переведенного текста - значение CIIM-экземпляра **translated_text.**

Примечание — Элемент **translatable_ text** определен в разделе F.4.1.6 ИСО 13584-42:2010.

В таблице F.7 приведены тексты и переведенные тексты, а также соответствующие им CIIM EXPRESS-экземпляры.

Таблица F.7 — Отображение OntoML-текста и переведенного текста

OntoML-текст	EXPRESS-экземпляры
<...> <text> a text </text> </...>	TEXT('a label') CIIM-представление непереведенной метки является строкой специфического типа данных **LABEL**.
<...> <text language="en"> a text </label> <text language="fr"> un texte </text> </...>	#tlabel=TRANSLATED_TEXT(('a text, 'un texte'), #pt); #pt=PRESENT_TRANSLATIONS((#lc1, #lc2)); #lc1=LANGUAGE_CODE ('en', $); #lc2=LANGUAGE_CODE('fr', $); CIIM-представление переведенного текста состоит из экземпляров следующих трех объектных CIIM EXPRESS-типов данных: **translated_text, present_translations** and **language_code**

F.4.3.4.2.6 Отображение синонимических имен

Функция **createSynonymous** позволяет создавать CIIM EXPRESS-ресурсы, соответствующие некоторой информации о синонимических метках, возможно, переведенных. Соответствующая запись имеет следующий вид:

createSynonymous(ontoMLLabel: XPath): SET OF syn_name_type

где:

— *ontoMLLabel:* XPath, который ссылается на OntoML-множество XML-элементов **label** (возможно, связанного с XML-атрибутом **language**), предназначенных для обработки;

— *syn_name_type:* общий тип данных, выдаваемый путем запроса этой функции. В случае непереведенной метки она выдает значение объектного CIIM-экземпляра **label**, а в случае переведенной метки - значение CIIM-экземпляра **label_with_language**.

Примечание — Элемент **syn_name_type** определен в разделе F.3.9.1.16 ИСО 13584-42:2010.

В таблице F.8 приведены синонимические имена и переведенные синонимические имена, а также соответствующие им CIIM EXPRESS-экземпляры.

Таблица F.8 — Отображение OntoML-синонимов и переведенных синонимов

OntoML-синоним	EXPRESS-экземпляры
<…> <label> a synonymous </label> <label> another synonymous </label> </…>	[LABEL('a synonymous'), LABEL('another synonymous')] CIIM-представлением непереведенных синонимических имен является набор строк, чьим специфическим типом данных является тип **LABEL.**
<…> <label language="en"> a synonymous </label> <label language="fr"> un synonyme </label> <label language="fr"> un autre synonyme </label> </…>	#syn1=LABEL_WITH_LANGUAGE('a synonymous', #lc1); #syn2=LABEL_WITH_LANGUAGE('un synonyme', #lc2); # syn3 =LABEL_WITH_LANGUAGE('un autre synonyme', #lc2); #lc1=LANGUAGE_CODE ('en', $); #lc2=LANGUAGE_CODE('fr', $); CIIM-представление переведенной синонимической метки состоит из экземпляров следующих трех объектных CIIM EXPRESS-типов данных: **label_with_language** и **language_code**

F.4.3.4.2.7 Отображение ключевых слов

Функция **createKeywords** позволяет формировать CIIM EXPRESS-ресурсы, соответствующие определенной информации относительно меток ключевых слов (возможно, переведенных). Соответствующая запись имеет следующий вид:

createKeywords(ontoMLLabel: XPath): SET OF keyword_type

где:

— *ontoMLLabel:* XPath, который ссылается на OntoML-множество XML-элементов **label** (возможно, связанного с XML-атрибутом **language**), предназначенных для обработки;

— *keyword_type:* общий тип данных, выдаваемый путем запроса этой функции. В случае непереведенной метки она выдает значение объектного CIIM-экземпляра **label,** а в случае переведенной метки – значение CIIM-экземпляра **label_with_language.**

Примечание — Элемент **keyword_type** определен в разделе F.3.9.1.17 ИСО 13584-42:2010.

В таблице F.9 приведены ключевые слова и переведенные ключевые слова, а также соответствующие им CIIM EXPRESS-экземпляры.

Таблица F.9 — Отображение OntoML-ключевых слов и переведенных ключевых слов

OntoML-ключевое слово	EXPRESS-экземпляры
<…> <label> a keyword </label> <label> another keyword </label> </…>	[LABEL('a keyword'), LABEL('another keyword')] CIIM-представление непереводимых ключевых слов – это совокупность строк, чьим специфическим типом данных является **LABEL**
<…> <label language="en"> a keyword </label> <label language="fr"> un mot cle </label> <label language="fr"> un autre mot cle </label> </…>	#syn1=LABEL_WITH_LANGUAGE('a keyword', 'en') ; #syn2=LABEL_WITH_LANGUAGE('un mot cle', 'fr'); # syn3 =LABEL_WITH_LANGUAGE('un autre autre mot cle', ' f r ') ; #lc1=LANGUAGE_CODE ('en', $); #lc2=LANGUAGE_CODE('fr', $); CIIM-представление переведенного ключевого слова состоит из экземпляров следующих двух объектных CIIM EXPRESS-типов данных: **label_with_language** и **language_code.**

F.4.3.4.2.8 Отображение документации

Связь документации с онтологическими CIIM-понятиями может осуществлятьмя тремя способами, а именно:

— либо путем ссылки на документ, представляемый как экземпляр онтологического CIIM-понятия: в этом случае будут использоваться объектные CIIM EXPRESS-элементы **referenced_graphics** и **referenced_document** entities are used;

— либо путем ссылки на хорошо идентифицированный документ: в этом случае будет использоваться объектный CIIM EXPRESS-элемент **identified_document**;

— либо путем ссылки на http-ресурс: в этом случае будут использоваться объектные CIIM EXPRESS-элементы **external_graphics, document_content, message, illustration, a6_illustration** и **a9_illustration**;

Для этой последней функциональной группы протокол и переведенная информация должна предоставляться в соответствии с CIIM-моделью. Эти CIIM EXPRESS-ресурсы могут классифицироваться в соответствии с CIIM-моделью следующим образом:

— графические материалы: EXPRESS-ресурсы, которые представляют собой CIIM-ресурс **graphics:** элементы **referenced_graphics** и **external_graphics;**

— документы: EXPRESS-ресурсы, которые представляют собой CIIM-ресурс **document:** элементы **referenced_document** и **identified_document;**

— спецификация на содержание онтологического понятия документа: EXPRESS-ресурсы, которые представляют собой CIIM-ресурс: элемент **document_content;**

— внешние ресурсы расширения класса: EXPRESS-ресурсы, которые представляют собой CIIM-ресурс: элементы **class_extension_external_item: message, illustration, a6_uillustration** и **a9_illustration.**

Отображающие функции определяются в соответствии с вышеприведенной классификацией следующим образом:

— отображающая функция, предназначенная для создания графических материалов:

createGraphics(ontoMLGraphicsType: XPath): graphics,

где:

— *ontoMLGraphicsType:* XPath, который ссылается на атрибут ***xs[i]:type*** элемента, для которого эта функция применима; она позволяет определять, какой вид графических материалов необходимо создать;

— *graphics:* тип объектных данных для экземпляра класса, выдаваемый путем запроса этой функции; если функция применяется к XML-элементу, чья модель содержания определена как ссылочная графика (**xsi:type = ontoml:REFERENCED_GRAPHICS_Type**), то она выдает значение объектного CIIM-экземпляра **referenced_graphics**; если же функция применяется к XML-элементу, чья модель содержания определена как внешняя графика (**xsi:type = ontoml:EXTERNAL_GRAPHICS_Type**), то она выдает значение объектного CIIM-экземпляра **external_graphics**;

— отображающая функция, предназначенная для создания документа:

createDocument(ontoMLDocumentType: XPath): document,

где:

— *ontoMLDocumentType:* XPath, который ссылается на атрибут ***xs[i]:type*** элемента, для которого эта функция применима; она позволяет определять, какой вид документа необходимо создать;

— *document:* тип объектных данных для экземпляра класса, выдаваемый путем запроса этой функции; если эта функция применяется к XML-элементу, чья модель содержания определена как ссылочный документ **(xsi:type = ontoml:REFERENCED_DOCUMENT_Type),** то она выдает значение объектного CIIM-экземпляра **referenced_document**; если же эта функция применяется к XML-элементу, чья модель содержания определена как идентифицированный документ **(xsi:type = ontoml:IDENTIFIED_DOCUMENT_Type),** то она выдает значение объектного CIIM-экземпляра **identified_document**;

— отображающая функция, предназначенная для создания спецификации на содержание онтологических понятий документа:

createDocumentContent(doc: string): document_content,

где:

— *doc:* XPath, который позволяет извлекать идентификатор OntoML-документа, для которого определяется его содержание;

— *document_content:* тип объектных данных для экземпляра класса, выдаваемый путем запроса этой функции;

— отображающая функция, предназначенная для создания внешних ресурсов для расширений класса:

createExtResource(ontoMLExtResourceType: XPath): class_extension_external_item,

где:

— *ontoMLExtResourceType:* XPath, который ссылается на атрибут ***xs[i]:type*** элемента, для которого эта функция применима; она позволяет определять, какой вид документа необходимо создать;

— *class_extension_external_item:* тип объектных данных для экземпляра класса, выдаваемый путем запроса этой функции; если функция применяется к XML-элементу, чья модель содержания определена как сообщение **(xsi:type = MESSAGE_Type),** то она выдает значение объектного CIIM-экземпляра **message**; если эта функция применяется к XML-элементу, чья модель содержания определена как иллюстрация **(xsi:type = ILLUSTRATION_Type),** в которой отсутствует XML-арибут **standard_size,** то она выдает значение объектного CIIM-экземпляра **illustration**; если эта функция применяется к XML-элементу, чья модель содержания определена как иллюстрация **(xsi:type = ILLUSTRATION_Type),** в которой определен XML-атрибут **standard_size**, то она выдает значение объектного CIIM-экземпляра **a6_illustration,** если этим XML-атрибутом является **a6_illustration** или **a9_illustration;**

— Указанные выше четыре отображающие функции предназначены для обработки частей OntoML-документа, который состоит из древа XML-субэлементов, чьим корнем является OntoML-элемент, для которого вызываются эти отображающие функции.

Отображающие функции, создающие объектные экземпляры CIIM-типов данных **external_graphics, document_content, message, illustration, a6_illustration** и **a9_illustration,** имеют дело с ссылкой на http-ресурсы. В соответствии с CIIM-моделью для каждой из этих функций необходимо создавать следующие объектные экземпляры, соответствующие представлению HTTP-протокола (см. таблицу F.10).

Таблица F.10 — Отображение OntoML HTTP-протокола

OntoML-представление	EXPRESS-экземпляры
Отсутствие точной информации в OntoML-языке.	#http_prot_name=ITEM_NAMES(LABEL('Hype rtext Transfer Protocol'), () , LABEL('HTTP/1.1'), $, $); #org=ORGANIZATION('', 'World Wide Web Consortium', 'W3C'); #http_protocol=HTTP_PROTOCOL(#org, 'USA' , 'NONE' , '001' , $, #http_prot_name, $, 2621) ; Экземпляр **http_protocol** предназначен для ссылок на него

Отображающие функции, формирующие CIIM-экземпляры типов данных **external_graphics, document_content, message, illustration, a6_illustration** и **a9_illustration,** также рассматриваются вместе с отображениями. В соответствии с CIIM-моделью необходимо для каждой из указанных выше функций сформировать следующие экземпляры элементов, соответствующие представлению отображений внешних ресурсов (см. таблицу F.11).

Таблица F.11 — Преобразуемые и непреобразуемые отображения OntoML-файлов

OntoML-представление	EXPRESS-экземпляры
<file> <file_name>filename.ext </file_name> <dir_name>directory name</dir_name> </file> **dir_name** – это дополнительный OntoML-элемент.	#ext_cont=NOT_TRANSLATED_EXTERNAL_CONTENT((#lang_spec_cont)); #lang_spec_cont=LANGUAGE_SPECIFIC_CONTENT((#http_file), #http_file, 'utf-8') ; #http_file=HTTP_FILE('filename.ext', $, mime_type, mime_subtype, $, 'filename.ext', # http_class_dir, $); Экземпляры ***mime_type*** и ***mime_subtype*** предназначены для расчета из имени файла. #http_class_dir=HTTP_CLASS_DIRECTORY ('directory_name', #class); Если имя каталога не указывается, то его необходимо создать и контролировать с помощью программы отображения.

Окончание таблицы F.11

OntoML-представление	EXPRESS-экземпляры
<file language="en"> <file_name>filename_en.ext </file_name> <dir_name>directory_name_1 </dir_name> </file> <file language="fr"> <file_name>filename_fr.ext </file_name> <dir_name>directory_name_2 </dir_name> </file> **dir_name** – это дополнительный OntoML-элемент.	#pt=PRESENT_TRANSLATIONS((#lc1, #lc2)); #lc1=LANGUAGE_CODE ('en', $); #lc2=LANGUAGE_CODE('fr', $); #lang_spec_cont_1=LANGUAGE_SPECIFIC_CO NTENT((#http_file_1), #http_file_1, "utf-8"); #lang_spec_cont_2=LANGUAGE_SPECIFIC_CO NTENT((#http_file_2), #http_file_2, 'utf-8'); #http_file_1=HTTP_FILE('***filename_en.ex*** **t'** , $, mime_type, mime_subtype, $, '[1]filename_en.ext', # http_class_dir_1, $); #http_file_2=HTTP_FILE('***filename_fr.ex*** **t'** , $, mime_type, mime_subtype, $, 'filename fr.ext', # http class dir 2, $); Экземпляры ***mime_type*** и ***mime_subtype*** предназначены для расчета из имени файла. #http_class_dir_1=HTTP_CLASS_DIRECTORY ("directory_name_1", #class); #http_class_dir_2=HTTP_CLASS_DIRECTORY ("directory_name_2", #class); Если имя каталога не указывается, то его необходимо создать и контролировать с помощью программы отображения

В таблице F.12 для каждого из компонентов OntoML-документа дается соответствующее CIIM EXPRESS-представление. Когда это необходимо, дается ссылка на ранее введенные CIIM EXPRESS-экземпляры объектов.

Таблица F.12 — Отображение (преобразование) внешних OntoML-ресурсов

OntoML-представление	EXPRESS-экземпляры
< ... xsi : type= **"REFERENCED_GRAPHICS_Type">** <graphics_reference document ref="documentId"/> <.../>	#ref_graph=REFERENCED_GRAPHICS(#doc); Экземпляр ***#doc*** формируется из OntoML-компонента ***documentId***, (см. п. F.4.3.4.2.1). Функция **Graphics** позволяет обращаться к экземпляру ***#ref_graph***
< ... xsi : type= **"REFERENCED_DOCUMENT_Type">** <document_reference document_ref="documentId"/> <.../>	#ref_doc=REFERENCED_DOCUMENT(#doc); Экземпляр ***#doc*** формируется из OntoML-компонента ***documentId***, (см. п. F.4.3.4.2.1). Функция **createDocument** позволяет обращаться к экземпляру ***#ref_doc***
< ... xsi : type= **"IDENTIFIED_DOCUMENT_Type">** <document_identifier> anIdentifier </document_identifier> <.../>	#doc_id=IDENTIFIED_DOCUMENT(<anIdentif ier>) Функция **createDocument** позволяет обращаться к экземпляру элемента ***#doc_id***

Продолжение таблицы F.12

OntoML-представление	EXPRESS-экземпляры
< ... xsi : type= **"EXTERNAL_GRAPHICS_Type">** <file> ... </file> … <file> ... </file> <…/>	#ext_graph=EXTERNAL_GRAPHICS(#graph_files); #graph_files=GRAPHIC_FILES(#http_proto col, #ext_cont); Экземпляр ***#http_protocol*** определяет HTTP-протокол Экземпляр ***#ext_cont*** определяет допустимые преобразования Функция **createGraphics** позволяет обращаться к экземпляру ***#ext_graph***
< ... xsi : type= **"DOCUMENT_CONTENT_Type">** <file> ... </file> … <file> ... </file> <revision>rev</revision> <…/>	#doc_cont=DOCUMENT_CONTENT(#doc, #http_protocol, #ext_cont, rev); Экземпляр ***#doc*** представляет собой экземпляр идентификатора понятия онтологии документа (см. п. F.4.3.4.2.1) Экземпляр ***#http_protocol*** определяет HTTP-протокол Экземпляр ***#ext_cont*** определяет допустимые преобразования Функция **createDocumentContent** позволяет обращаться к экземпляру ***#doc_cont***

Продолжение таблицы F.12

OntoML-представление	EXPRESS-экземпляры
< ... xsi : type= **"MESSAGE_Type">** <file> ... </file> … <file> ... </file> <code>a code</code> <…/>	#mess=MESSAGE(#http_protocol, #ext_cont, <aCode>); Экземпляр ***#http_protocol*** определяет HTTP-протокол Экземпляр ***#ext_content*** определяет допустимые преобразования. Функция **createExtResource** позволяет обращаться к экземпляру элемента ***#mess***
< ... xsi : type= **"ILLUSTRATION_Type">** <file> ... </file> … <code>aCode</code> <kind_of_content> *aKind*</kind_of_content> <width> a*Width*</width> <height>aHeight</height> <…/>	#ill = ILLUSTRATION(#http_protocol, #ext_cont, <aCode>, <aKind>, #width, #height); #width=LENGTH_MEASURE_WITH_UNIT(LENGTH _MEASURE(<aWidth>), #lu_width); #lu_width=(LENGTH_UNIT()SI_UNIT(.MILLI ., .METRE.)); #height=LENGTH_MEASURE_WITH_UNIT(LENGTH_MEASURE(<aHeighth>), #lu_height); #lu_height=(LENGTH_UNIT()SI_UNIT(.MILL I., .METRE.)); Экземпляр ***#http_protocol*** определяет HTTP-протокол Экземпляр ***#ext_content*** определяет допустимые преобразования. Функция **createExtResource** позволяет обращаться к экземпляру ***#ill***

Окончание таблицы F.12

OntoML-представление	EXPRESS-экземпляры
< ... xsi : type= **"ILLUSTRATION_Type"** standard_size="**a6_illustration**"> <file> ... </file> … <code>aCode</code> <kind_of_content> *aKind*</kind_of_content> <width> a*Width*</width> <height>aHeight</height> <…/>	#a6ill=A6_ILLUSTRATION(#http_protocol, #ext_cont, <aCode>, <aKind>, #width, #height); Экземпляр ***#http_protocol*** определяет HTTP-протокол Экземпляр ***#ext_content*** определяет допустимые преобразования. Экземпляры ***#width*** и ***#height*** определены выше. Функция **createExtResource** позволяет обращаться к экземпляру ***#a6ill***
< ... xsi : type= **"ILLUSTRATION_Type"** standard_size="**a9_illustration**"> <file> ... </file> … <code>aCode</code> <kind_of_content> *aKind*</kind_of_content> <width> a*Width*</width> <height>aHeight</height> <…/>	#a9ill=A9_ILLUSTRATION(#http_protocol, #ext_cont, <aCode>, <aKind>, #width, #height); Экземпляр ***#http_protocol*** определяет HTTP-протокол Экземпляр ***#ext_content*** определяет допустимые преобразования. Экземпляры ***#width*** и ***#height*** определены выше. Функция **createExtResource** позволяет обращаться к экземпляру ***#a9ill***

F.4.3.4.2.9 Пример *апостериорного* преобразования соотношений

OntoML-представление *апостериорных* семантических соотношений не идентично используемому в CIIM EXPRESS-модели. Соответственно, преобразование может выражаться только с помощью соответствующей функции преобразования. Запись этой функции имеет следующий вид:

createAPosteriori(ontoMLAposteriori: XPath): a_posteriori_semantic_relationship

где:

— ***ontoMLAposteriori***: значение XPath, которое определяет атрибут **xsi:type** элемента, для которого применима данная функция; оно позволяет определять, какой вид *апостериорных* семантических соотношений предназначен для его создания;

— ***a_posteriori_semantic_relationship***: тип объектных данных экземпляра, возвращаемый путем обращения к этой функции; если эта функция применима к XML-элементу, чья модель содержания определена как *апостериорный* тип семантического соотношения **(xsi:type = ontoml:A_POSTERIORI_CASE_OF_Type)**, то она будет обращаться к значению CIIM- экземпляра объекта **a_posteriori_case_of**; если эта функция применима к XML-элементу, чья модель содержания определена как *апостериорный* тип семантического соотношения **(xsi:type = ontoml:A_POSTERIORI_VIEW_OF_Type)**, то она будет обращаться к значению CIIM- экземпляра

объекта **a_posteriori_view_of**.

В таблице F.13 представлен пример OntoML- *апостериорных* представлений и соответствующие им CIIM EXPRESS-представления.

Таблица F.13 — Пример *апостериорного* OntoML-преобразования соотношений

OntoML-представление	EXPRESS-экземпляры
< ... xsi : type= **"A_POSTERIORI_CASE_OF_Type">** <source class_ref="classId1"/> <is_case_of class_ref="classId2"/> <corresponding_properties> <mapping> <domain> <property property_re f="propId1"> </domain> <range property_ref="propId2"/> </mapping> </corresponding_properties> <.../>	#apcof=A_POSTERIORI_CASE_OF(#cl1, #cl2, ((#prop1, #prop2))); Экземпляры **#cl1** и ***#cl2*** – это соответственно сформированные с использованием OntoML-языка идентификаторы ***classId1*** и ***classId2*** *(см. п. F.4.3.4.2.1.* Экземпляры ***#prop1*** и ***#prop2*** - это соответственно сформированные с использованием OntoML-языка идентификаторы ***propId1*** и ***propId2*** согласно п. *F.4.3.4.2.1.* Функция **createAPosteriori** позволяет обращаться к экземпляру ***#apcof***.

В таблице F.14 представлен *апостериорный* вид OntoML-представлений и соответствующие им CIIM EXPRESS-представления.

Таблица F.14 — OntoML- *апостериорное* преобразование производных отношений

OntoML-представление	EXPRESS-экземпляры
< ... xsi : type= **"A_POSTERIORI_VIEW_OF_Type">** <functional_model class_ref="classId1"/> <is_view_of class_ref="classId2"/> <corresponding_properties> <mapping> <domain> <property property_ref="propId1"> </domain> <range property_ref="propId2"/> </mapping> </corresponding_properties> <.../>	#apcof=A_POSTERIORI_VIEW_OF(#cl1, #cl2, ((#prop1, #prop2))); Экземпляры **#cl1** и ***#cl2*** - это соответственно сформированные с использованием OntoML-языка идентификаторы ***classId1*** и ***classId2*** согласно п. *F.4.3.4.2.1.* Экземпляры ***#prop1*** и ***#prop2*** - это соответственно сформированные с использованием OntoML-языка идентификаторы ***propId1*** *и* ***propId2*** согласно п. *F.4.3.4.2.1.* Функция **createAPosteriori** позволяет обращаться к экземпляру *#apcof.*

F.4.3.4.2.10 Отображение экземпляров элементов

OntoML-язык не задает какую-либо структуру для представления значений и экземпляров, однако он использует ресурсы, указанные в формате обмена продукцией (см. ИСО/ТС 29002-10).

Отображение значений свойств и экземпляров класса выходит за рамки рассмотрения OntoML-языка, однако для обеспечения непротиворечивости предоставляются две абстрактные функции преобразования.

Функция **createValue** позволяет преобразовывать значения, определенные в формате обмена общими продуктами, в соответствующий экземпляр (экземпляры) CIIM-элемента. Запись этой функции имеет следующий вид:

createValue(OntoMLValues: XPath): LIST OF primitive_value,

где:

— ***ontoMLValues***: значение XPath, которое определяет множество XML-элементов **value,** предназначенных для обработки.

Примечание 1 — XML-элемент **value,** определенный в формате обмена информацией о продукции, рассмотрен в ИСО/ТС 29002-10.

— ***property_value***: общий тип данных, возвращаемых путем обращения к данной функции, которая представляет собой совместимое с CIIM представление значения.

Примечание 2 — Элемент **primitive_value** рассмотрен в разделе 6.3.2 ИСО 13584-24:2003.

Функция **createPopulation** позволяет преобразовывать экземпляры класса, определенные в их обобщенном формате для соответствующего экземпляра (экземпляров) CIIM-элемента. Запись этой функции имеет следующий вид:

createPopulation(OntoMLInstances: XPath): LIST OF UNIQUE dic_class_intance,

— ***ontoMLInstances***: значение XPath, которое определяет множество XML-элементов **item,** предназначенных для обработки.

Примечание 3 — XML-элемент **item,** определенный в формате обмена информацией о продукции, рассмотрен в ИСО/ТС 29002-10.

— ***dic_class_instance***: общий тип данных, возвращаемых путем обращения к данной функции, которая является совместимым с CIIM- представлением экземпляра.

Примечание 4 — Элемент **dic_class_instance** рассмотрен в разделе 6.4.7.1 ИСО 13584-24:2003.

F.4.4 Отображение OntoML-ресурсов для CIIM EXPRESS-элементов, на которые отсутствуют ссылки

На OntoML-языке могут представляться некоторые CIIM EXPRESS-ресурсы, на которые отсутствуют ссылки. Таким образом, преобразование не может выражаться обычным способом, то есть на основе некоторых понятий CIIM-онтологии или на основе общей структуры каталога.

Для этой цели вместо определения функций, результат действия которых предназначен для присвоения целевого EXPRESS-пути, определяется ряд процедур.

F.4.4.1 Онтология глобального языка

Процедура **create_global_language_assignment** используется для создания CIIM EXPRESS-экземпляра типа элемента данных **GLOBAL_LANGUAGE_ASSIGNMENT** из определенного языка. Запись этой процедуры имеет следующий вид:

create_global_language_assignment(language_id : XPath, country_id : XPath),

где:

— ***language_id***: значение XPath, воспроизводящее значение атрибута на обобщенном OntoML-языке;

— ***country_id***: XPath, который воспроизводит значение атрибута на обобщенном OntoML-языке.

В таблице F.15 приведен пример применения этой процедуры (при этом предполагается, что процедура преобразования относится к атрибуту XML-языка для онтологического XML-элемента, т.е.:

create_global_language_assignment(@language_code, @country_code)).

Таблица F.15

OntoML	EXPRESS-экземпляр
<ontoml> <global_language language_code="en"/> </ontoml>	# gla=GLOBAL_LANGUAGE_AS SIGNMENT ('en', $);

F.5 Таблица соответствия между комплексными типами OntoML-данных и типами CIIM EXPRESS-данных

В таблице F.16 показано соответствие между полной совокупностью конкретных комплексных типов OntoML-данных и типами CIIM EXPRESS-конструкций (элементов или данных).

Таблица F.16 — Соответствие между комплексными типами OntoML-данных и типами CIIM EXPRESS-данных

Комплексный тип OntoML-данных	Элемент CIIM EXPRESS-данных
A_POSTERIORI_CASE_OF_Type	a_posteriori_case_of
A_POSTERIORI_SEMANTIC_RELATIONSHIP_Type	a_posteriori_semantic_relationship
A_POSTERIORI_SEMANTIC_RELATIONSHIPS_Type	LIST_of _a_posteriori_semantic_relationship
A_POSTERIORI_VIEW_OF_Type	a_posteriori_view_of
ALTERNATIVE_UNIT_IDS_Type	LIST[1:?]_of _dic_unit_identifier
ALTERNATIVE_UNITS_Type	LIST[1:?] of _dic_unit
ANY_TYPE_Type	data_type
ARRAY_TYPE_Type	array_type
AUTHORS_Type	LIST[1:?]_ of _person
AXIS1_PLACEMENT_TYPE_Type	axis1_placement_type
AXIS2_PLACEMENT_2D_TYPE_Type	axis2_placement_2d_type
AXIS2_PLACEMENT_3D_TYPE_Type	axis2_placement_3d_type
BAG_TYPE_Type	bag_type
BOOLEAN_TYPE_Type	boolean_type
CARDINALITY_CONSTRAINT_Type	cardinality_constraint
CATEGORIZATION_CLASS_Type	categorization_class
CLASS_CONSTANT_VALUES_Type	SET OF class_value_assigment
CLASS_CONSTRAINT_Type	class_constraint
CLASS_EXTENSION_Type	class_extension
CLASS_REFERENCE_TYPE_Type	class_reference_type
CLASS_PRESENTATION_ON_PAPER_Type	LIST_of _illustration
CLASS_PRESENTATION_ON_SCREEN_Type	LIST_of _illustration
CLASS_Type	class
CLASS_VALUE_ASSIGNMENT_Type	class_value_assignment
CLASSIFICATION_Type	SET OF property_classification
CONDITION_DET_Type	condition_DET
CONFIGURATION_CONTROL_CONSTRAINT_Type	configuration_control_constraint

Продолжение таблицы F.16

Комплексный тип OntoML-данных	Элемент CIIM EXPRESS-данных
CONSTRAINT_OR_CONSTRAINT_ID_Type	constraint_or_constraint_id
CONSTRAINT_Type	constraint
CONSTRAINTS_Type	SET OF constraint_or_constraint_id
CONTAINED_CLASS_EXTENSIONS_Type	SET OF class_extension
CONTAINED_CLASSES_Type	SET OF class
CONTAINED_DATATYPES_Type	SET OF data_type_element
CONTAINED_DOCUMENTS_Type	SET OF document_element
CONTAINED_PROPERTIES_Type	SET OF property
CONTAINED_SUPPLIERS_Type	SET OF supllier_element
CONTEXT_DEPENDENT_UNIT_Type	context_dependent_unit
CONTEXT_PARAM_ICON_Type	LIST OF a6_illustration
CONTEXT_PARAMETER_CONSTRAINTS_Type	SET OF property_constraint
CONTEXT_RESTRICTION_CONSTRAINT_Type	context_restriction_constraint
CONVERSION_BASED_UNIT_Type	conversion_based_unit
CORRESPONDING_PROPERTIES_type	SET OF LIST[2:2] OF property_BSU
CREATE_ICON_Type	LIST OF a6_illustration
DATATYPE_Type	data_type_element
DATE_DATA_TYPE_Type	date_data_type
DATE_TI M E_DATA_TYPE_Type	date_time_data_type
DEPENDENT_P_DET_Type	dependent_P_DET
DERIVED_UNIT_ELEMENT_Type	derived_unit_element
DERIVED_UNIT_Type	derived_unit
DIC_UNIT_REFERENCE_Type	dic_unit_identifier
DIC_UNITS_REFERENCE_Type	SET[1:?] OF dic_unit_identifier
DIC_UNIT_Type	dic_unit
DIC_VALUE_Type	dic_value
DICTIONARY_IN_STANDARD_FORMAT_Type	dictionary_in_standard_format
DICTIONARY_Type	dictionary
DIMENSIONAL_EXPONENTS_Type	dimensional_exponents
DOCUMENT_CONTENT_Type	document_content
DOCUMENT_IDENTIFIER_NAME_LABEL_Type	source_doc_type
DOCUMENT_IDENTIFIER_Type	document_identifier
DOCUMENT_Type	document
DOMAIN_CONSTRAINT_Type	domain_constraint
ENUMERATION_CONSTRAINT_Type	enumeration_constraint
EXPLICIT_FUNCTIONAL_MODEL_CLASS_EXTENSION_Type	explicit_functional_model_class_extension
EXPLICIT_ITEM_CLASS_EXTENSION_Type	explicit_item_class_extension
EXTERNAL_GRAPHICS_Type	external_graphics
EXTERNAL_RESOURCE_Type	external_content
FILTER_Type	filter
FM_CLASS_VI EW_OF_Type	fm_class_view_of
FUNCTIONAL_MODEL_CLASS_Type	functional_model_class
G EN ERAL_TEXT_Type	text or translated_text
EXTERNAL_FILES_Type	external_item
GRAPHICS_Type	graphics
HEADER_Type	
HTTP_FILE_Type	http_file

IDENTIFIED_DOCUMENT_Type	identified_document
ILLUSTRATION_Type	illustration
INFORMATION_Type	
INT_CURRENCY_TYPE_Type	int_currency_type
1 NT_D I C_VALU E_Type	dic_value
INT_MEASURE_TYPE_Type	int_measure_type
INT_TYPE_Type	int_type
INTEGRITY_CONSTRAINT_Type	integrity_constraint
ITEM_CLASS_CASE_OF_Type	item_class_case_of
ITEM_CLASS_Type	item_class
ITS_VALUES_Type	LIST OF dic_value
LANGUAGE_Type	language_code
LEVEL_Type	level
LEVEL_TYPE_Type	level_type
LIBRARY_IIM_IDENTIFICATION_Type	library_iim_identification
LIBRARY_IN_STANDARD_FORMAT_Type	library_in_standard_format
LIBRARY_Type	library
LIST_TYPE_Type	li°t_type
MAPPING_FUNCTION_Type	
MATHEMATICAL_STRING_Type	mathematical_string
MESSAGE_Type	message
NAMED_TYPE_Type	named_type
NAMED_UNIT_Type	named_unit
NON_DEPENDENT_P_DET_Type	non_dependent_P_DET
NON_INSTANTIABLE_FUNCTIONAL_VIEW_CLASS_ Type	non_instanciable_functional_view_class
NON_QUANTITATIVE_CODE_TYPE_Type	non_quantitative_code_type
NON_QUANTITATIVE_INT_TYPE_Type	non_quantitative_int_type
NON_SI_UNIT_Type	non_si_unit
NON_TRANSLATABLE_STRING_TYPE_Type	non_translatable_string_type
NUMBER_TYPE_Type	number_type
ONTOML_Type	
ORGANIZATION_Type	organization
PERSON_Type	person
PLACEMENT_TYPE_Type	placement_type
POSTCONDITION_Type	SET[1:?] OF filter
PRECONDITION_Type	SET OF filter
PREFERRED_NAME_LABEL_Type	label or translated_label
PREFERRED_NAME_Type	translatable_label
PROGRAM_REFERENCE_TYPE_Type	program_reference_type
PROPERTY_CLASSIFICATION_Type	property_classification
PROPERTY_CONSTRAINT_Type	property_constraint
PROPERTY_MAPPING_Type	
PROPERTY_Type	property_DET
PROPERTY VALUE RECOMMENDED PRESENTA TION_Type	property_value_recommended_representation
RANGE_CONSTRAINT_Type	range_constraint
REAL_CURRENCY_TYPE_Type	real_currency_type
REAL_MEASURE_TYPE_Type	real_measure_type
REAL_TYPE_Type	real_type
RECOMMENDED_PRESENTATION_Type	SET OF property_value_recommended_representation
REFERENCED_DOCUMENT_Type	referenced_document
REFERENCED_GRAPHICS_Type	referenced_graphics
REMOTE_HTTP_ADDRESS_Type	remote_http_address
REMOTE_LOCATIONS_Type	LIST of absolute_URL_type

Окончание таблицы F.16

Комплексный тип OntoML-данных	Элемент CIIM EXPRESS-данных
REPRESENTATION_CONTEXT_Type	representation_context
REPRESENTATION_P_DET_Type	representation_P_DET
REPRESENTATION_REFERENCE_TYPE_Type	representation_reference_type
SET_TYPE_Type	set_type
SET_WITH_SUBSET_CONSTRAINT_TYPE_Type	set_with_subset_constraint
SHORT_NAME_LABEL_Type	label or translated_label
SHORT_NAME_Type	translatable_label
SI_UNIT_Type	SI_unit
SOURCE_DOCUMENT_Type	document
STRING_DIC_VALUE_Type	dic_value
STRING_PATTERN_CONSTRAINT_Type	string_pattern_constraint
STRING_SIZE_CONSTRAINT_Type	string_size_constraint
STRING_TYPE_Type	string_type
STRINGS_Type	SET OF STRING
SUBCLASS_CONSTRAINT_Type	subclass_constraint
SUBSET_Type	LIST[1:?] OF UNIQUE primitive_value
SUPPLIER_Type	supplier_element
SUPPORTED_VEP_Type	SET OF view_exchange_protocol_identification
SYNONYMOUS_NAME_LABEL_Type	syn_name_type
SYNONYMOUS_NAME_TYPE_Type	SET OF syn_name_type
SYNONYMOUS_SYMBOLS_Type	SET[1:?] OF mathematical_string
TEXT_Type	translatable_text
TI ME_DATA_TYPE_Type	time_data_type
TRANSLATABLE_STRING_TYPE_Type	translatable_string_type
TRANSLATION_DATA_Type	translation_data
TRANSLATION_Type	LIST OF translation_data
UNIT_ID_Type	dic_unit_identifier
UNIT_Type	unit
V_C_V_RANG E_Type	SET OF view_control_variable_range
VIEW_CONTROL_VARIABLE_RANGE_Type	view_control_variable_range
VIEW_EXCHANGE_PROTOCOL_IDENTIFICATION_ Type	view_exchange_protocol_identification

Приложение G

(справочное)

Уровни обмена экземплярами OntoML-документа

OntoML-язык обеспечивает XML-описания, которые позволяют как простым, так и усложненным онтологиям сочетаться с обобщенной моделью ИСО 13584/МЭК 61360, которой необходимо обмениваться с помощью XML.

Кроме того, OntoML-язык может использоваться в качестве коммуникативного формата при ответе на запросы, выполняемые с использованием механизма анализа понятий в каталогах согласно ИСО/ТС 29002-20. Для этой цели в OntoML-языке определена совокупность глобальных XML-элементов, позволяющих производить обмен теми экземплярами OntoML-документа, чей корневой элемент является одним из тех, что определен как глобальный XML-элемент.

Ниже перечислены эти глобальные XML-элементы:

— Глобальный элемент **ontoml:** Наиболее общий глобальный XML-элемент, который позволяет представлять экземпляры OntoML-документа для обмена ими в виде либо одиночной онтологии, либо одиночной библиотеки, либо онтологии со связанной с ней библиотекой.

Примечание 1 — Общая структура OntoML-языка рассмотрена в разделе 6.4.

— Глобальный элемент **supplier:** Позволяет представлять экземпляры OntoML-документа для обмена информационными элементами, описывающими поставщика, который идентифицируется IRDI-идентификатором.

Примечание 2 — Понятие онтологии поставщика рассмотрено в разделе 6.7.1.

Примечание 3 — Идентификация понятия онтологии поставщика рассмотрена в разделе 9.1.1.

— Глобальный элемент **class:** Позволяет представлять экземпляры OntoML-документа для обмена информационными элементами, описывающими класс, который идентифицируется IRDI-идентификатором.

Примечание 4 — Понятие онтологии класса рассмотрено в разделе 6.7.2 и 6.7.3.

Примечание 5 — Идентификатор онтологического понятия класса рассмотрен в разделе 9.1.3.1.

— Глобальный элемент **property:** Позволяет представлять экземпляры OntoML-документа для обмена информационными элементами, описывающими свойство (признак), которое идентифицируется IRDI- идентификатором.

Примечание 6 — Понятие онтологии свойства рассмотрено в разделах 6.7.4 и 6.7.5.

Примечание 7 — Идентификатор онтологического понятия свойства рассмотрен в разделе 9.1.3.2.

— Глобальный элемент **datatype:** Позволяет представлять экземпляры OntoML-документа для обмена информационными элементами, описывающими тип данных, который идентифицируется IRDI-идентификатором.

Примечание 8 — Понятие онтологии типа данных рассмотрено в разделе 6.7.6.

Примечание 9 — Идентификатор онтологического понятия типа данных рассмотрен в разделе 9.1.3.2.

— Глобальный элемент **document:** Позволяет представлять экземпляры OntoML-документа для обмена информационными элементами, описывающими документ, который идентифицируется IRD-идентификатором.

Примечание 10 — Онтологическое понятие документа рассмотрено в разделе 6.7.7.

Примечание 11 — Идентификатор онтологического понятия документа рассмотрен в разделе 9.1.3.2.

— Глобальный элемент **dic_unit:** Позволяет представлять экземпляры OntoML-документа для обмена информационными элементами, описывающими словарную единицу, которая идентифицируется IRDI- идентификатором.

Примечание 12 — Понятие словарной единицы рассмотрено в разделе 8.4.

Примечание 13 — Идентификатор онтологического понятия словарной единицы рассмотрен в разделе 9.1.3.3.

— Глобальный элемент **constraint:** Позволяет представлять экземпляры OntoML-документа для обмена информационными элементами, описывающими ограничительное условие, которое идентифицируется IRDI- идентификатором.

Примечание 14 — Понятие ограничительного условия рассмотрено в разделе 8.5.

Примечание 15 — Идентификатор онтологического понятия ограничительного условия рассмотрен в разделе 9.1.3.3.

— Глобальный элемент **dic_value:** Позволяет представлять экземпляры OntoML-документа для обмена информационными элементами, описывающими словарную единицу, которая идентифицируется IRDI- идентификатором.

Примечание 16 — Понятие словарного значения рассмотрено в разделе 8.3.4.

Примечание 17 — Идентификатор онтологического понятия словарного значения рассмотрен в разделе 9.1.3.3.

— Глобальный элемент **a_posteriori_semantic_relationship:** Позволяет представлять экземпляры OntoML-документа для обмена информационными элементами, описывающими *апостериорную* семантическую связь, которая идентифицируется IRDI-идентификатором.

Примечание 18 — Понятие *апостериорного* семантического соотношения рассмотрено в разделе 8.6.

Примечание 19 — Идентификатор онтологического понятия *апостериорного* семантического соотношения рассмотрен в разделе 9.1.3.3.

Приложение Н

(справочное)

Перечень форматов значений

ИСО 13584-32 устанавливает особый синтаксис для определения допустимых форматов записи строк и численных значений, которые могут связываться с некоторым свойством (признаком).

Пример 1 — Формат NR1 3 устанавливает, что допускается использование только целых значений, состоящих из трех цифр.

Примечание 1 — Никакой формат чисел не устанавливается для какого-либо элемента, кроме типа данных **ANY_TYPE_Type**, включая и тип данных **BOOLEAN_TYPE_Type**.

Примечание 2 — В ИСО 13584-32 задание формата для значения свойства не является обязательным.

Синтексис допустимых форматов указан в настоящем приложении с использованием варианта расширенной нормальной формы Бэкуса-Наура (EBNF), описанного в ИСО/МЭК 14977.

Пример 1 — Синтаксис формата NR1 3 – это литеры 'NR1" ' '3'.

Содержание каждого синтаксиса – это символы, которые могут использоваться для представления какого-либо значения, но не могут быть определены с помощью EBNF-формы, поэтому содержание каждой части формата, связанной с допустимым представлением значения, определяется отдельно для каждой части этого формата.

Пример 2 — Синтаксис формата NR1 3 имеет следующее содержание: NR1 означает, что допускается представление только в виде целого числа, пробел - что форматом определяется только фиксированное число символов, а цифра 3 – что это число символов должно быть равно трем.

Н.1 Обозначения

В таблице Н.1 приведена сводка вариантов синтаксического метаязыка EBNF-формы ИСО/МЭК 14977, используемых в ИСО 13584-32 для определения форматов значений свойств (признаков).

С использованием этих обозначений синтаксис варианта синтаксического метаязыка EBNF-формы, используемый в ИСО 13584-32 для определения форматов значений свойств, может быть подытожен с помощью следующей грамматики (символы мета-идентификатора, буквы и цифры не детализируются):

```
syntax = syntaxrule ,{ syntaxrule } ;
syntaxrule = metaidentifier , '=' , definitionslist , ';' ;
definitionslist = singledefinition , { '|' , singledefinition } ;
singledefinition = term , { ',', term } ;
term = primary, [ '-', primary } ;
primary = optionalsequence | repeatedsequence | groupedsequence |
          metaidentifier | terminal | empty ;
optionalsequence = '[' definitionslist ']' ;
repeatedsequence = '{' definitionslist '}' ;
groupedsequence = '(' definitionslist ')' ;
metaidentifier = letter , { letter } ;
terminal = "'", (character - "'"),{ character - "'" }, "'"
           | '"', (character - '"'),{ character - '"' }, '"' ;
empty = ;
```

Знак равенства '=' указывает правило синтаксиса. Мета-идентификатор слева может быть заменен на совокупность элементов из правой части. Любые пробелы, имеющиеся между элементами, не должны приниматься во внимание, если они находятся в пределах терминального символа. Синтаксическое правило ограничивается знаком ';'.

Таблица Н.1 — Синтаксический метаязык EBNF-формы ИСО/МЭК 14977

Представление	Наименование символа согласно ИСО/МЭК 10646-1	Символ метаязыка и его назначение
' '	Апостроф	Символ первой кавычки представляет собой терминальные символы языка. Этот символ не должен содержать апострофа. Пример: 'Hello'
"..."	Кавычки	Символ второй кавычки представляет собой терминальные символы языка. Этот символ не должен содержать апострофа. Пример: "John's car"
( )	Левая/правая круглые скобки	Начало/конец символов групп, содержимое которых должно рассматриваться как единый символ.
[ ]	Левая/правая квадратные скобки	Начало/конец вспомогательных символов, содержимое которых может существовать или не существовать
{ }	Левая/правая фигурные скобки	Начало/конец повторяющихся символов, содержимое которых может повторяться 0 - n – раз.
-	Дефис-минус	Символ вычитания
,	Запятая	Символ связи
=	Знак равенства	Символ определения Синтаксическое правило определяет символ слева с помощью формулы, стоящей справа от этого знака.
|	Вертикальный штрих	Символ альтернативного разделителя
;	Точка с запятой	Терминальный символ, обозначающий конец синтаксического правила

Применение мета-идентификатора в перечне определений означает нетерминальный символ, который появляется слева от другого синтаксического правила. Мета-идентификатор содержит буквы или цифры, но первым символом всегда является буква. Если член содержит знак «минус», перед которым и после которого стоит какой-либо символ, то только первый из этих символов будет представлять этот член.

Пример 1 — Обозначение вида:

" ' ", символ - " ' ", " ' "

означает любой символ, кроме символа апострофа, находящегося между двумя кавычками.

Терминальный символ означает тот символ, который не может расширяться с помощью синтаксического правила, и который будет проявляться в качестве окончательного результата. Для представления терминального символа допускаются два способа: либо с помощью множества символов без апострофов, вставляемых между двумя апострофами; либо с помощью множества символов без кавычек, вставляемых между двумя кавычками.

Пример 2 — Допустим, мы желаем описать с помощью подобной грамматики стоимость изделия в евро (€), которая выражается положительной величиной не более чем с двумя значащими цифрами в центах. Введем мета-идентификатор, связанный с тремя следующими синтаксическими правилами:

```
digit = '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9';
cents = [ '.' , digit [ , digit ] ];
euros = digit { , digit } cents;
```

При использовании этих синтаксических правил 012, 4323.3, 3.56 будут являться примерами допустимых представлений евро. 12., .10 – это примеры недопустимых представлений евро.

Н.2 Типы форматов значений данных

Грамматика, описанная в настоящем приложении, определяет девять различных типов форматов

значений: четыре – количественных формата, и еще пять - неколичественных формата.

В следующем разделе мы определим мета-идентификаторы, которые используются для задания этих форматов. В разделе H.4 определяется синтаксическое правило для четырех мета-идентификаторов, соответствующих четырем количественным форматам значений вместе с содержанием на количественном уровне.

H.3 Мета-идентификатор, используемый для определения форматов

Мета-идентификатор, используемый в грамматике, которая определяет различные форматы значений, имеет следующий вид:

```
dot = '.';
decimalMark = '.';
exponentIndicator = 'E';
numeratorIndicator = 'N';
denominatorIndicator = 'D';
leadingDigit = '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9';
lengthOfExponent = leadingDigit, {trailingDigit};
lengthOfIntegerPart = (leadingDigit, {trailingDigit});
lengthOfNumerator = leadingDigit, {trailingDigit};
lengthOfDenominator = leadingDigit, {trailingDigit};
lengthOfFractionalPart = (leadingDigit, {trailingDigit}) | '0' ;
lengthOfIntegralPart = (leadingDigit, {trailingDigit})| '0' ;
lengthOfNumber = leadingDigit, {trailingDigit};
trailingDigit = '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9';
signedExponent = 'S';
signedNumber = space,'S';
space = ' ';
variableLengthIndicator = '..';
```

где

decimalMark – это разделитель между целочисленной и дробной частями значений формата NR2 or NR3;

leadingDigit – первый шифр номера, содержащего один или несколько других шифров;

trailingDigit – один из шифров, который составляется для формирования значений, за исключением первого.

Примечание — Если этот номер содержит лишь одну цифру, то компонент trailingDigit будет отсутствовать.

H.4 Форматы количественных значений

Синтаксические правила для четырех форматов количественных значений и их содержание, используемое для представления этих значений, определены в четырех подклассах, в которых допускается использование свойств следующих типов данных:

— Типа **NUMBER_TYPE_Type, INT_TYPE_Type, INT_MEASURE_TYPE_Type, INT_CURRENCY_TYPE_Type, NON_QUANTITATIVE_INT_TYPE_Type, REAL_TYPE_Type, REAL_MEASURE_TYPE_Type, REAL_CURRENCY_TYPE_Type, RATIONAL_TYPE_Type** или **RATIONAL_MEASURE_TYPE_Type;**

— Типа **LEVEL_TYPE_Type,** чьим элементом **value_type** является **NUMBER_TYPE_Type, INT_TYPE_Type, INT_MEASURE_TYPE_Type, INT_CURRENCY_TYPE_Type, NON_QUANTITATIVE_INT_TYPE_Type, REAL_TYPE_Type, REAL_MEASURE_TYPE_Type, REAL_CURRENCY_TYPE_Type, RATIONAL_TYPE_Type** или **RATIONAL_MEASURE_TYPE_Type;**

— Типа **LIST_TYPE_Type, SET_TYPE_Type, BAG_TYPE_Type, ARRAY_TYPE_Type or SET_WITH_SUBSET_CONSTRAINT_TYPE_Type,** чьим элементом **value_type** является **NUMBER_TYPE_Type, INT_TYPE_Type, INT_MEASURE_TYPE_Type, INT_CURRENCY_TYPE_Type, NON_QUANTITATIVE_INT_TYPE_Type, REAL_TYPE_Type, REAL_MEASURE_TYPE_Type, REAL_CURRENCY_TYPE_Type, RATIONAL_TYPE_Type** или **RATIONAL_MEASURE_TYPE_Type.**

Примечание 1 — Для типа данных **NON_QUANTITATIVE_INT_TYPE_Type** формат значений применим к кодам.

Примечание 2 — Значение данного атрибута должно быть совместимым с типом данных для свойства; оно не должно изменять этот тип данных; в противном случае оно должно игнорироваться.

Пример — Формат значений NR2 несовместим с типом данных INT_TYPE_Type, поскольку целые значения не должны иметь дробной части.

H.4.1 Формат значений NR1

Синтаксис для формата значений NR1 определяет целые значения какого-либо свойства.

Синтаксическое правило:

NR1Value = 'NR1', ((signedNumber, variableLengthIndicator) | (signedNumber, space) | variableLengthIndicator | space), lengthOfNumber;

Содержание компонентов для формата значений NR1 при представлении этих значений таково:

— ' NR1 ': Значение должно быть целым.

Примечание 1 — Числовые значения в формате NR1 не должны содержать пробелов.

— Компонент **lengthOfNumber**: Определяет число цифр в значении.

Примечание 2 — Если этот компонент находится перед компонентом **variableLengthIndicator**, то фактическое число цифр может быть меньшим.

— Компонент **signedNumber**: При наличии компонента **signedNumber** связанное с ним число должно быть положительным, отрицательным или нулевым. При наличии положительных значений номера перед ними могут стоять знаки '+', а перед отрицательными номерами – знаки '-'. При нулевом значении знак '-' не ставится.

— Компонент **variableLengthIndicator**: При наличии компонента **variableLengthIndicator** связанный с ним номер должен содержать такое число цифр, которое не будет превышать таковое в спецификации длины, т.е. **lengthOfNumber**.

H.4.2 Формат значений NR2

Синтаксис для формата значений NR2 определяет действительные значения свойства, которое не требует показателя степени.

Синтаксическое правило:

NR2Value = 'NR2', ((signedNumber,variableLengthIndicator) | (signedNumber, space) | variableLengthIndicator space), lengthOfIntegralPart, decimalMark, lengthOfFractionalPart;

Содержание компонентов для формата значений NR2 при представлении этих значений таково:

— ' NR2 ': Значение должно быть действительным.

Примечание 1 — Числовые значения в формате NR2 не должны содержать пробелов.

— Компонент **lengthOfFractionalPart**: Число цифр в дробной части числа.

Примечание 2 — Если этот компонент находится перед компонентом **variableLengthIndicator**, то фактическое число цифр может быть меньшим.

Примечание 3 — Компонент **lengthOfFractionalPart** однозначно задает рекомендуемую точность определения значения. Реальная точность числа, из которого было получено данное значение, может быть выше выражаемого в этом компоненте.

— Компонент **lengthOfIntegralPart**: Определяет число цифр в целой части числа.

Примечание 4 — Перед компонентом **variableLengthIndicator** фактическое число цифр может быть меньшим.

— Компонент **signedNumber**: При наличии компонента **signedNumber** связанное с ним число должно быть положительным, отрицательным или нулевым. При наличии положительных значений номера перед ними могут стоять знаки '+', а перед отрицательными номерами – знаки '-'. При нулевом значении знак '-' не ставится.

— Компонент **variableLengthIndicator**: При наличии компонента **variableLengthIndicator** связанный с ним номер должен содержать такое число цифр, которое не будет превышать таковое в спецификации длины, т.е. в **lengthOfIntegralPart** или **lengthOfFractionalPart**. В этом числе должен содержаться по крайней мере один шифр.

H.4.3 Формат значений NR3

Синтаксис для формата значений NR3 определяет действительные значения какого-либо свойства, которые представляются вместе с показателем степени.

Синтаксическое правило:

NR3Value = 'NR3', ((signedNumber, variableLengthIndicator) (signedNumber, space) | variableLengthIndicator | space), lengthOfIntegralPart, decimalMark, lengthOfFractionalPart, exponentIndicator, [signedExponent], lengthOfExponent;

Содержание компонентов для формата значений NR3 при представлении этих значений таково:

— ' NR3 ': Значение должно быть действительным, вместе с показателем степени при

основании 10.

Примечание 1 — Мантисса числа должна содержать по крайней мере одну цифру и десятичную точку, а показатель экспоненты – также не менее одной цифры.

Примечание 2 — Числовые значения в формате NR3 не должны содержать пробелов.

— Компонент **exponentIndicator**: Является разделителем между мантиссой и экспонентой числа в числах в формате NR3.

— Компонент **lengthOfExponent**: Определяет число цифр в экспоненте.

Примечание 3 — Если этот компонент находится перед компонентом **variableLengthIndicator**, то фактическое число цифр может быть меньшим.

Примечание 4 — В общем случае существующие знак числа или десятичная точка не учитываются в компоненте **lengthOfNumber, lengthOfIntegralPart, lengthOfFractionalPart** или **lengthOfExponent**.

— Компонент **lengthOfFractionalPart**: Определяет число цифр в дробной части мантиссы.

Примечание 5 — Если этот компонент находится перед компонентом **variableLengthIndicator**, то фактическое число цифр может быть меньшим.

Примечание 6 — Компонент **lengthOfFractionalPart** однозначно задает рекомендуемую точность определения значения. Реальная точность числа, из которого было получено данное значение, может быть выше выражаемого в этом компоненте.

— Компонент **lengthOfIntegralPart**: Определяет число цифр в целой части мантиссы.

Примечание 7 — Если этот компонент находится перед компонентом **variableLengthIndicator**, то фактическое число цифр может быть меньшим.

— Компонент **signedExponent**: При наличии компонента **signedNumber** связанное с ним число должно быть положительным, отрицательным или нулевым. При наличии положительных значений номеров перед ними могут стоять знаки '+', а перед отрицательными номерами – знаки '-'. При нулевом значении знак '-' не ставится.

— Компонент **variableLengthIndicator**: При наличии компонента **variableLengthIndicator** связанное с ним число должно содержать такое число цифр, которое не будет превышать таковое в спецификации длины, т.е. в **lengthOfIntegralPart, lengthOfFractionalPart** или **lengthOfExponent**. В этой мантиссе и экспоненте должно содержаться по крайней мере по одному шифру.

H.4.4 Формат значений NR4

Синтаксис для формата значений NR4 определяет рациональные значения какого-либо свойства, которые представляются вместе с их целыми (и, возможно, дробными) частями вместе с числителями и знаменателями.

Синтаксическое правило:

NR4Value = 'NR4', ((signedNumber,variableLengthIndicator) | (signedNumber, space) | variableLengthIndicator | space) , lengthOfIntegerPart, numeratorIndicator, lengthOfNumerator, denominatorIndicator, lengthOfDenominator;

Содержание компонентов для формата значений NR4 при представлении этих значений таково:

— ' NR4 ': Значение должно быть рациональным, представляемым в виде либо целочисленного, либо дробного значения и содержащим числитель и знаменатель (или целую и дробную часть).

Пример — Значения $12^{1}/_{2}$ и $12^{3}/_{4}$ могут представляться в формате NR4.

Примечание 1 — В целой части числа (или в числителе и знаменателе) должна иметься по крайней мере одна цифра. Если одна часть содержит цифру, то другая часть также должна содержать несколько цифр. Все три части также могут содержать цифры.

Примечание 2 — Числовые значения в формате NR4 не должны содержать пробелов.

— Компонент **numeratorIndicator**: Является разделителем между описанием целой и дробной частей числа в формате NR4.

— Компонент **lengthOfNumerator**: Определяет число цифр в числителе.

Примечание 3 — Если этот компонент находится перед компонентом **variableLengthIndicator**, то фактическое число цифр может быть меньшим.

Примечание 4 — Если значение рационального числа полностью характеризуется его целой частью, то ни числитель дроби, ни ее знаменатель не должны представляться.

— Компонент **denominatorIndicator**: Является разделителем между описаниями числителя и знаменателя в форматах NR4.

— Компонент **lengthOf Denominator**: Определяет число цифр в знаменателе.

Примечание 5 — Если этот компонент находится перед компонентом **variableLengthIndicator**, то фактическое число цифр может быть меньшим.

Примечание 6 — Если значение рационального числа полностью характеризуется его целой частью, то ни числитель дроби, ни ее знаменатель не должны представляться.

— Компонент **lengthOfIntegerPart**: Определяет число цифр в целой части рационального числа.

Примечание 7 — Если этот компонент находится перед компонентом **variableLengthIndicator**, то фактическое число цифр может быть меньшим.

— Компонент **variableLengthIndicator**: При наличии компонента **variableLengthIndicator** связанное с ним число должно содержать такое число цифр, которое не будет превышать таковое в спецификации длины, т.е. в компонентах **lengthOfIntegralPart**, **lengthOfNumerator** или **lengthOfDenominator**. В целой части числа или в двух частях дроби должны содержаться по крайней мере по одному шифру.

H.5 Форматы неколичественных значений

Синтаксические правила для четырех форматов неколичественных значений и их содержание, используемое для представления этих значений, определены в пяти подклассах, в которых допускается использование свойств следующих типов данных.

— Типа **STRING_TYPE_Type**, **TRANSLATABLE_STRING_TYPE_Type**, **NON_TRANSLATABLE_STRING_TYPE_Type**, **URI_TYPE_Type**, **NON_QUANTITATIVE_CODE_TYPE_Type**, **DATE_DATA_TYPE_Type**, **DATE_TIME_TYPE_Type** или **TI M E_DATA_TYPE_Type**;

— Типа **LEVEL_TYPE_Type**, чьим элементом **value_type** является **STRING_TYPE_Type**, **TRANSLATABLE_STRING_TYPE_Type**, **NON_TRANSLATABLE_STRING_TYPE_Type**, **URI_TYPE_Type**, **NON_QUANTITATIVE_CODE_TYPE_Type**, **DATE_DATA_TYPE_Type**, **DATE_TIME_TYPE_Type** или **TIME_DATA_TYPE_Type;**

— Типа **LIST_TYPE_Type**, **SET_TYPE_Type**, **BAG_TYPE_Type**, **ARRAY_TYPE_Type** или **SET_WITH_SUBSET_CONSTRAINT_TYPE_Type**, чьим элементом **value_type** является **STRING_TYPE_Type**, **TRANSLATABLE_STRING_TYPE_Type**, **NON_TRANSLATABLE_STRING_TYPE_Type**, **URI_TYPE_Type**, **NON_QUANTITATIVE_CODE_TYPE_Type**, **DATE_DATA_TYPE_Type**, **DATE_TIME_TYPE_Type** или **TIME_DATA_TYPE_Type.**

Примечание 1 — Значение данного атрибута должно быть совместимым с типом данных для свойства; оно не должно изменять этот тип данных, в противном случае оно должно игнорироваться.

Примечание 2 — Для типа данных **NON_QUANTITATIVE_CODE_TYPE_Type** формат значений применим к кодам.

Неколичественные значения представляются с помощью строк, содержащих символы, причем длина этих строк может либо непосредственно определяться верхним пределом содержащихся символов, либо путем задания того общего числа символов, которым может быть любое целочисленное кратное значение определенной длины.

Синтаксическое правило:

factor (индекс) = leadingDigit, {trailingDigit};

Компонент **numberOfCharacters** = (leadingDigit, {trailingDigit})|('(nx' , factor,')');

Содержание компонентов **factor** таково:

— Компонент **factor**: При наличии индекса компонент **numberOf Characters** может быть любым целым числом, кратным значению индекса. Последний не должен содержать нулевого значения.

— Компонент **numberOfCharacters**: Определяет максимальное количество символов, содержащихся в строке.

H.5.1 Формат буквенных значений

Формат Alphabetic Value Format (A) определяет значения в виде строки, которая будет содержать буквенные символы, поэтому ее содержимое должно выбираться из символов строки 00, ячеек 20, 40 – 7E или ячеек C0 – FF основного многоязычного уровня (Basic Multilingual Plane, BMP) (Уровень 00 Группы 00) в ИСО/МЭК 10646-1.

Примечание 1 — Из-за возможных проблем с интерпретацией содержания значения в компонентах одной или нескольких систем рекомендуется, когда это возможно, ограничивать используемые символы множеством G0, указанном в ИСО/МЭК 10646-1, и/или строкой 00 столбцов 002 – 007, указанной в ИСО/МЭК 10646-1.

Примечание 2 — Для альтернативных языков, поддерживаемых переведенными данными, доступны соответствующие символы или идеограммы связанного с данным языком множества специфических символов, как это определено стандартом на уникод (Unicode Standard). В большинстве случаев не существует соотношения 1:1 между символами исходного языка и символами объектного языка.

Примечание 3 — В большинстве случаев не существует соотношения 1:1 между символами исходного языка и символами объектного языка.

Пример — CJK - Китайские, японские и корейские идеограммы (иероглифы).

Синтаксическое правило:

AValue = 'A', (space | variableLengthIndicator), numberOfCharacters;

Содержание компонентов A-формата значений для их представления таково:

— Компонент ' **A** ': Значение должно быть строкой или несколькими частями строки, состоящими из буквенных символов.

— Компонент **variableLengthIndicator**: При наличии компонента **variableLengthIndicator** строка может содержать меньшее число символов, чем это указано в компоненте **numberOf Characters**. Строка должна содержать по крайней мере один символ.

H.5.2 Формат смешанных символов значений

Формат Mixed Value Format (M) определяет значения строки, которая будет содержать любой из символов, указанных в разделе H.7.

Примечание — Для альтернативных языков, поддерживаемых переведенными данными, доступны соответствующие символы или идеограммы связанного с данным языком множества специфических символов, как это определено стандартом на уникод (Unicode Standard).

Пример — CJK - Китайские, японские и корейские символы (иероглифы).

Синтаксическое правило:

MValue = 'M', (space | variableLengthIndicator), numberOfCharacters;

Содержание компонентов значений в M-формате для их представления таково:

— Компонент '**M**': Значение должно быть строкой или несколькими частями строки.

— Компонент **variableLengthIndicator:** При наличии компонента **variableLengthIndicator** строка может содержать меньшее число символов, чем это указано в компоненте numberOf Characters. Строка должна содержать по крайней мере один символ.

H.5.3 Формат числовых хначений

Формат Number Value Format (N) определяет формат значений в виде строки, которая будет содержать исключительно цифры, поэтому ее содержание должно выбираться из символов строки 00, ячеек 2B, 2D, 30 – 39 или ячейки 45 основного многоязычного уровня (Basic Multilingual Plane, BMP) (Уровень 00 Группы 00) в ИСО/МЭК 10646-1.

Примечание — Для альтернативных языков, поддерживаемых переведенными данными, доступны соответствующие символы или идеограммы связанного с данным языком множества специфических символов, как это определено стандартом на уникод (Unicode Standard).

Пример — В таблице H.2 представлен результат преобразования европейских цифр от "0" до "9" в арабские цифры.

Таблица H.2 — Таблица соответствия европейских цифр арабским

European digits	9	8	7	6	5	4	3	2	1	0
Arabic digits	٩	٨	٧	٦	٥	٤	٣	٢	١	٠

Надписи в таблице: 1 – Европейские цифры; 2 – Арабские цифры.

Синтаксическое правило:

NValue = 'N', (space | variableLengthIndicator), numberOfCharacters;

Содержание компонентов N-формата значений для их представления таково:

— Компонент ' N ': Значение должно быть строкой или несколькими частями строки, состоящими из цифр;

— Компонент **variableLengthIndicator**: При наличии компонента **variableLengthIndicator** строка может содержать меньшее число символов, чем это указано в компоненте **numberOf Characters**. Строка должна содержать по крайней мере один символ.

H.5.4 Формат смешанных буквенных и цифровых значений

Формат Mixed Alphabetic or Numeric Characters Value Format (X) определяет значения строки,

которая будет содержать буквенно-цифровые символы, т.е. любую комбинацию символов, входящих в A- и N-форматы значений.

Примечание — Для альтернативных языков, поддерживаемых переведенными данными, доступны соответствующие символы или идеограммы связанного с данным языком множества специфических символов, как это определено стандартом на уникод (Unicode Standard).

Синтаксическое правило:
XValue = 'X', (space | variableLengthIndicator), numberOfCharacters;
Содержание компонентов значений в X-формате для их представления таково:
— Компонент '**X**': Значение должно быть строкой или несколькими частями строки из букв, цифр или их сочетания;
— Компонент **variableLengthIndicator**: При наличии компонента **variableLengthIndicator** строка может содержать меньшее число символов, чем это указано в компоненте **numberOf Characters**. Строка должна содержать по крайней мере один символ.

H.5.5 Формат двоичных значений

Формат Binary Value Format (B) определяет значения строки, которая будет содержать двоичные символы, то есть "0" или "1", поэтому ее содержание должно выбираться из символов строки 00, ячеек 30, 31 основного многоязычного уровня (Basic Multilingual Plane, BMP) (Уровень 00 Группы 00) в ИСО/МЭК 10646-1.

Примечание — Для альтернативных языков, поддерживаемых переведенными данными, доступны соответствующие символы или идеограммы связанного с данным языком множества специфических символов, как это определено стандартом на уникод (Unicode Standard).

Синтаксическое правило:
BValue = 'B', (space | variableLengthIndicator), numberOfCharacters;
Содержание компонентов B-формата значений для их представления таково:
— Компонент ' **B**': Значение должно быть строкой или несколькими частями строки, состоящими из символа "0" или "1" (или последовательностей этих символов).
— Компонент **variableLengthIndicator**: При наличии компонента **variableLengthIndicator** строка может содержать меньшее число символов, чем это указано в компоненте **numberOf Characters**. Строка должна содержать по крайней мере один символ.

H.6 Примеры количественных значений

В нижеприведенной таблице H.3 содержится ряд примеров величин, которые могут содержаться в номерах и строках, характеризуемых приведенной выше структурой формата значений.

Таблица H.3 — Примеры количественных значений

Формат значения	Пример допустимых значений
NR1 3	123; 001; 000;
NR1..3	123; 87; 5
NR1S3	+123;+000;
NR1S..3	-123; +1; 0; -12;
NR2 3.3	123.300; 000.400; 000.420;
NR2..3.3	321.233; 1.234; 23.56; 9.783; .72; 324.
NR2S 3.3	-123.123;+123.300;
NR2S..3.3	-123.123;+12.3; 0.1;+.4;-3. ; 0. ; .0;
NR3 3.3E4	123.123E0004; 003.000E1000;
NR3 3.3ES4	123.123E+0004; 123.123E0004; 123.000E-0001
NR3..3.3E4	123.123E0004; .123E0001; 5.E1234
NR3S 3.3ES4	+123.123E+0004; 123.000E-0001;
NR3S..3.3ES4	-123.123E+0004; +1.00E-01; .0E0; +3.E-1; -.2E-1000;
NR4 3N2D2	001 02/03; 012 00/01; 123 03/04; 000 01/04

Окончание таблицы Н.1

Формат значения	Пример допустимых значений
NR4..3N2D2	1 $^{1}/_{2}$; 12; 123 $^{3}/_{4}$; $^{1}/_{4}$;
NR4S 3N2D2	+001 02/03; -012 00/01; 123 03/04; -000 01/04
NR4S..3N2D2	-1 $^{1}/_{2}$; 12; +123 $^{3}/_{4}$; $^{1}/_{4}$;
A19	My name is Reinhard, abcdefghijklmnopqrs
A..3	Abc; de; G
X..5	A23RN1;B1;ca
M..10	A23RN1; B1; ca. 256 urn;
N (nx5)	12345; 1234512345; 222223333344444;
N..(nx5)	1234; 2345; 34512345; 1234512345; 23333344444; 222223333344444; -3; 5E2;
B1	0; 1;
B3	011; 101;

Н.7 Символы, заимствованные из ИСО/МЭК 10646-1

Нижеперечисленные символы должны использоваться в формате Mixed Value Format (M) (см. п. Н.5.2):

— все символы из строки 00 основного многоязычного уровня (Basic Multilingual Plane, BMP) (Уровень 00 Группы 00) в ИСО/МЭК 10646-1;

— символы из других строк указанного уровня в ИСО/МЭК 10646-1 (см. таблицу Н.4);

— для других языков, поддерживаемых переводными данными, полный набор символов, определенный в стандарте на уникод.

Примечание 1 — Из-за возможных проблем с интерпретацией смысла содержания в компонентах одной или нескольких систем рекомендуется, когда это возможно, регистрировать используемые символы в наборе G0 ИСО/МЭК 10646-1 и/или в строке 00 столбцов 002 - 007 ИСО/МЭК 10646-1.

Примечание 2 — Стандарт на уникод опубликован Unicode Consortium, P.O. Box 391476, Mountain View, CA 94039-1476, U.S.A., www.unicode.org.

Таблица Н.4 — Символы из других строк основного многоязычного уровня ИСО/МЭК 10646-1

Символ	Наименование	Столбец	Ячейка
-	ТИРЕ	02	C7
≡	ИДЕНТИЧНОСТЬ	22	61
∧	ЛОГИЧЕСКОЕ «И»	22	27
∨	ЛОГИЧЕСКОЕ «ИЛИ»	22	28
∩	ПЕРЕСЕЧЕНИЕ	22	29
∪	ОБЪЕДИНЕНИЕ	22	2A
⊂	ПОДМНОЖЕСТВО ОТ (ЗАКЛЮЧЕННОГО)	22	82
⊃	ПОДМНОЖЕСТВО ОТ (СОДЕРЖИМОГО)	22	83
⇐	ЛЕВАЯ ДВОЙНАЯ СТРЕЛКА (ВЫТЕКАЮЩАЯ ИЗ)	21	D0
⇒	ПРАВАЯ ДВОЙНАЯ СТРЕЛКА (ЗАКЛЮЧАЮЩАЯ)	21	D2
∴	СЛЕДСТВИЕ	22	34
∵	ПРИЧИНА	22	35
∈	ЭЛЕМЕНТ В...	22	08
∋	СОДЕРЖАНИЕ КАК ЧЛЕНА (НАЛИЧИЕ КАК ЭЛЕМЕНТА)	22	0B
⊆	ПОДМНОЖЕСТВО ИЛИ РАВЕНСТВО .... (СОДЕРЖАНИЕ КАК ПОДКЛАССА)	22	86
⊇	НАДМНОЖЕСТВО ИЛИ РАВЕНСТВО (СОДЕРЖАНИЕ КАК НАДКЛАССА)	22	87

Продолжение таблицы Н.4

Символ	Наименование	Столбец	Ячейка
∫	ИНТЕГРАЛ	22	2B
∮	КОНТУРНЫЙ ИНТЕГРАЛ	22	2E
∞	БЕСКОНЕЧНОСТЬ	22	1E
∇	ОПЕРАТОР «НАБЛА»	22	07
∂	ЧАСТНАЯ ПРОИЗВОДНАЯ	22	02
~	ОПЕРАТОР «ТИЛЬДА» (РАЗНОСТЬ МЕЖДУ…)	22	3C
≈	ПРИБЛИЗИТЕЛЬНО РАВНО …	22	48
≡	АСИМПТОТИЧЕСКИ РАВНО …	22	43
≅	ПРИМЕРНО РАВНО … (АНАЛОГИЧНО…)	22	45
≤	МЕНЬШЕ ИЛИ РАВНО …	22	64
≠ <	НЕ РАВНО …	22	60
≥	БОЛЬШЕ ИЛИ РАВНО …	22	65
⇔	ЛЕВАЯ/ПРАВАЯ ДВОЙНАЯ СТРЕЛКА (ЕСЛИ И ТОЛЬКО ЕСЛИ)	21	D4
¬	ОТСУТСТВИЕ ЗНАКА	00	AC
∀	ДЛЯ ВСЕХ …	22	00
∃	СУЩЕСТВУЕТ	22	03
ℵ	БУКВА «АЛЕФ» НА ИВРИТЕ	05	D0
□	ОПЕРАТОР Д'АЛАМБЕРА)	25	A1
∥	ПАРАЛЛЕЛЬНОСТЬ …	22	25
Γ	ЗАГЛАВНАЯ ГРЕЧЕСКАЯ БУКВА «ГАММА»	03	93
Δ	ЗАГЛАВНАЯ ГРЕЧЕСКАЯ БУКВА «ДЕЛЬТА»	03	94
⊥	ПЕРПЕНДИКУЛЯР К …	22	A5
∠	УГОЛ	22	20
⊾	ПРЯМОЙ УГОЛ С ДУГОЙ	22	BE
Θ	ЗАГЛАВНАЯ ГРЕЧЕСКАЯ БУКВА «ТЭТА»	03	98

Продолжение таблицы Н.4

Символ	Наименование	Столбец	Ячейка
〈	LEFT POINTING ANGLE BRACKET (BRA)	23	29
〉	ПРАВАЯ УГЛОВАЯ СКОБКА	23	2A
Λ	ЗАГЛАВНАЯ ГРЕЧЕСКАЯ БУКВА «ЛЯМБДА»	03	9B
'	ШТРИХ	20	32
"	ДВОЙНОЙ ШТРИХ	20	33
Ξ	ЗАГЛАВНАЯ ГРЕЧЕСКАЯ БУКВА «КСИ»	03	9E
∓	ЗНАК «МИНУС ИЛИ ПЛЮС»	22	13
Π	ЗАГЛАВНАЯ ГРЕЧЕСКАЯ БУКВА «ПИ»	03	A0
²	ВЕРХНИЙ ИНДЕКС «ДВА»	00	B2
Σ	ЗАГЛАВНАЯ ГРЕЧЕСКАЯ БУКВА «СИГМА»	03	A3
×	ЗНАК УМНОЖЕНИЯ	00	D7
³	ВЕРХНИЙ ИНДЕКС «ТРИ»	00	B3
Υ	ЗАГЛАВНАЯ ГРЕЧЕСКАЯ БУКВА «ИПСИЛОН»	03	A5
Φ	ЗАГЛАВНАЯ ГРЕЧЕСКАЯ БУКВА «ФИ»	03	A6
·	СРЕДНЯЯ ТОЧКА	00	B7
Ψ	ЗАГЛАВНАЯ ГРЕЧЕСКАЯ БУКВА «ПСИ»	03	A8
Ω	ЗАГЛАВНАЯ ГРЕЧЕСКАЯ БУКВА «ОМЕГА»	03	A9
∅	НУЛЬ-МНОЖЕСТВО	22	05
⃗	ПРАВЫЙ ЗНАК ГАРПУНА С ВЕРХНИМ ЗУБЦОМ (ЧЕРТА НАД ВЕКТОРОМ)	21	C0
√	КОРЕНЬ КВАДРАТНЫЙ ОТ … (РАДИКАЛ)	22	1A
f	СТРОЧНАЯ ЛАТИНСКАЯ БУКВА «ЭФ» С КРЮЧКОМ (ФУНКЦИЯ ОТ …)	01	92
∝	ПРОПОРЦИОНАЛЬНОСТЬ …	22	1D
±	ЗНАК «ПЛЮС/МИНУС»	00	B1
°	ЗНАК ГРАДУСА	00	B0
α	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «АЛЬФА»	03	B1
β	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «БЕТА»	03	B2
γ	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «ГАММА»	03	B3
δ	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «ДЕЛЬТА»	03	B4
ε	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «ЭПСИЛОН»	03	B5
ζ	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «ДЗЕТА»	03	B6
η	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «ЭТА»	03	B7
θ	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «ТЭТА»	03	B8
ι	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «ИОТА»	03	B9
κ	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «КАППА»	03	BA
λ	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «ЛЯМБДА»	03	BB
μ	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «МЮ»	03	BC
ν	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «НЮ»	03	BD
ξ	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «КСИ»	03	BE
‰	ЗНАК ПРОМИЛЛЕ	20	30
π	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «ПИ»	03	C0
ρ	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «РО»	03	C1

Окончание таблицы Н.4

Символ	Наименование	Столбец	Ячейка
σ	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «СИГМА»	03	C3
÷	ЗНАК ДЕЛЕНИЯ	03	F7
τ	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «ТАУ»	03	C4
υ	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «ИПСИЛОН»	03	C5
φ	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «ФИ»	03	C6
χ	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «ХИ»	03	C7
ψ	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «ПСИ»	03	C8
ω	СТРОЧНАЯ ГРЕЧЕСКАЯ БУКВА «ОМЕГА»	03	C9
†	ЗНАК ССЫЛКИ	20	20
←	ЛЕВАЯ СТРЕЛКА	21	90
↑	ВЕРХНЯЯ СТРЕЛКА	21	91
→	ПРАВАЯ СТРЕЛКА	21	92
↓	НИЖНЯЯ СТРЕЛКА	21	93
¯	ВЕРХНЯЯ ЧЕРТА	20	3E

Приложение I
(справочное)
Пример XML-файла

В настоящем приложении в качестве примера приведены краткие сведения относительно различных ресурсов и этапов в описании библиотеки деталей и указанных в серии ИСО 13584.

Содержание данного раздела таково:

— иллюстративный пример;

— представление данного примера на OntoML-языке.

I.1 Иллюстративный пример

Рисунок I.1 иллюстрирует приведенный ниже пример. Существует класс характеристик кольцевых прокладок, обозначаемый как *paw*, которые продаются производителями подшипников и используются в некоторых механических конструкциях. Каждое основное понятие онтологии изделия представляется на единственном языке: английском.

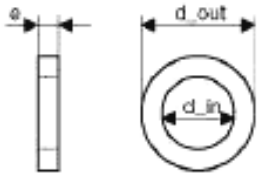


Рисунок I.1 — Обобщенный модельный пример: онтологическое определение

Данный класс характеристик изделия описывается тремя следующими отличительными признаками:

— внутренний диаметр кольца (символ *d_in*) - реальная мера, выражаемая в миллиметрах;

— внешний диаметр кольца (символ *d_out*) - реальная мера, выражаемая в миллиметрах;

— толщина кольца (символ *e*) - реальная мера, выражаемая в миллиметрах.

Данный класс характеристик продукции также связывается с чертежом (представленным, в частности, на рисунке I.1), называемым *paw.jpg*.

В рамках данного примера мы предлагаем определять следующую очень простую отнологию продукции:

— класс характеристик изделия, который играет роль исходного элемента онтологии изделия и чьим именем является *bearing;* этот класс определяется и описывается двумя свойствами:

— внутренний диаметр *(d_in);*

— внешний диаметр *(d_out).*

— класс характеристик изделия, соответствующий классу характеристик *paw*, подклассу класса *bearing*; он определяется и описывется единственным свойством:

— толщина (e).

Понятия CIIM–онтологии, необходимые для использования в данном примере, таковы: каталог, поставщик, класс и признак, для каждого из которых мы определяем (в соответствии со структурой идентификаторов, указанных в разделе 9.1) следующие идентификаторы:

— поставщик изделия: "0002-38491502100024";

— каталог: "0002-38491502100024#11-DICTIONARY#1";

— класс характеристик *bearing*: "0002-38491502100024#01-BEARING#1";

— класс характеристик *paw*: "0002-38491502100024#01-PAW#1";

— признак *inner diameter*: "0002-38491502100024#02-INNER_DIAMETER#1";

— признак *outer diameter*: "0002-38491502100024#02-OUTER_DIAMTER#1";

— признак *thickness*: "0002-38491502100024#02-THICKNESS#1".

В соответствии с данной структурой онтологии мы предлагаем описывать группу из пяти следующих изделий *paw* (см. рисунок I.2):

d_in	e	d_out
10	1	15
11	1	16.5
13	2	19.5
17	3	25.5
19	4	28.5

Рисунок I.2 — Общий модельный пример: технические характеристики изделия

Примечание — OntoML-язык предоставляет не только ресурсы для представления информации относительно структуры содержимого изделий (расширение класса характеристик изделия). Описание изделий как ссылки на признак и пару введенных значений выходит за рамки данного стандарта.

I.2 Представление на OntoML-языке

В этом случае полное представление примера изделия на OntoML-языке дается в разделе I.1. Данный XML-файл в соответствии с классом 3 данной части ИСО 13584 является матрицей отображения.

```
<?xml version="1.0" encoding="UTF-8"?>
<ontoml:ontoml xmlns:ontoml="urn:iso:std:iso:13584:-32:ed-1:tech:xml-schema:ontoml"
xmlns:cat="urn:iso:std:iso:ts:29002:-10:ed-1:tech:xml-schema:catalogue" xmlns:val="urn:iso:std:iso:ts:29002:-10:ed-
1:tech:xml-schema:value" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:iso:std:iso:13584:-32:ed-1:tech:xml-schema:ontoml
D:\plib\plibDoc\P32\IS\xml_schema\ontoml.xsd">
    <header id="0002-38491502100024#11-DICTIONARY#1">
        <global_language language_code="en"/>
        <description>This file illustrates the use of OntoML</description>
        <version>1</version>
        <name>generalModel.xml</name>
        <date_time_stamp>2010-04-07T08:11:49+02:00</date_time_stamp>
        <author>Eric SARDET</author>
```

```
        <organisation>My Company</organisation>
        <pre_processor_version/>
        <originating_system>XMLSpy</originating_system>
        <authorisation>Eric Sardet - sardet@ensma.fr</authorisation>
        <!-- The XML file is compliant with ISO13584-32, conformance class 3. The library_structure XML element
specifies this particular conformance class. -->
        <ontoml_information>
            <!--We assume that the dictionary/library revision is equal to 1-->
            <revision>1</revision>
            <preferred_name>
                <label>OntoML dictionary + library basic example</label>
            </preferred_name>
        </ontoml_information>
        <ontoml_structure>
            <status>IS</status>
            <name>ONTOML</name>
            <date>2010</date>
            <application>3</application>
        </ontoml_structure>
    </header>
    <!-- The ontology is defined using the dictionary XML element. Because it represents both the ontology itself and
instances compliant to this ontology, the specific content model used is based on the
LIBRARY_IN_STANDARD_FORMAT_Type complex type definition. An identifier is assigned to the dictionary using the id
XML attribute. -->
    <dictionary xsi:type="ontoml:DICTIONARY_IN_STANDARD_FORMAT_Type">
        <!--We assume that the information described is a complete library-->
        <is_complete>true</is_complete>
        <!-- The supplier responsible for the description of the ontology data is identified using the supplier_ref XML
attribute -->
        <responsible_supplier supplier_ref="0002-38491502100024"/>
        <!-- All the characterization classes described in an ontology are gathered in the "contained_classes" XML
element. -->
        <contained_classes>
            <!-- This class XML element defines the "bearing" class. It is defined as an item class ("ITEM_CLASS_Type"
complex type). It is identified using the "id" XML attribute.-->
            <ontoml:class xsi:type="ontoml:ITEM_CLASS_Type" id="0002-38491502100024#01-BEARING#1">
                <!-- We assume that the provided characterizationclass revision is equal to 1 -->
                <revision>1</revision>
                <!-- This characterization class preferred name is not translated and set to "bearing" -->
                <preferred_name>
                    <label>bearing</label>
                </preferred_name>
                <!-- The "bearing" characterization class definition is not translated. -->
                <definition>
                    <text>general bearing parts family</text>
                </definition>
                <!-- The "bearing" characterization class is described by two properties, each of them being identified
unambiguously ("property_ref" XMLattribute). In this example, the properties definitions are provided, but it is not
mandatory. The listed properties are: inner diameter and outer diameter.-->
                <described_by>
                    <property property_ref="0002-38491502100024#02-INNER_DIAMETER#1"/>
                    <property property_ref="0002-38491502100024#02-OUTER_DIAMETER#1"/>
                </described_by>
            </ontoml:class>
            <!-- This class XML element defines the "paw" class. It is defined as an item class ("ITEM_CLASS_Type"
complex type). It is identified using the id XML attribute.-->
            <ontoml:class xsi:type="ontoml:ITEM_CLASS_Type" id="0002-38491502100024#01-PAW#1">
                <!-- We assume that the provided characterization class revision is equal to "1" -->
                <revision>1</revision>
                <!-- This characterization class preferred name is not translated and set to "paw" -->
                <preferred_name>
                    <label>paw</label>
                </preferred_name>
                <!-- The "paw" characterization class definition is not translated. -->
                <definition>
                    <text>paw parts family</text>
                </definition>
```

```
            <!-- The "paw" characterization class is a subclass of the "bearing" characterization class  it is specified
using the "its_superclass" element whose "class_ref" attibute references the "bearing" characterization class-->
            <its_superclass class_ref="0002-38491502100024#01-BEARING#1"/>
            <!-- The "paw" characterization class is described by a single property being identified unambiguously
("property_ref" XMLattribute). In this example, the properties definitions are provided, but it is not mandatory. The listed
properties are: inner diameter and outer diameter -->
            <described_by>
                <property property_ref="0002-38491502100024#02-THICKNESS#001"/>
            </described_by>
            <!-- A drawing, stored in a file called "paw.jpeg" located in the same directory that the current XML file, is
assigned to the "paw" characterization class.-->
            <simplified_drawing xsi:type="ontoml:EXTERNAL_GRAPHICS_Type">
                <!-- The drawing consists of a single external resource called "paw.jpg"-->
                <representation>
                    <file>
                        <http_file>paw.jpeg</http_file>
                    </file>
                </representation>
            </simplified_drawing>
        </ontoml:class>
    </contained_classes>
    <!-- A name, that is only defined in english, is provided to the dictionary -->
    <!--Every supplier referenced in this ontology shall be described in the "contained_suppliers" XML element. -->
    <contained_suppliers>
        <ontoml:supplier id="0002-38491502100024">
            <revision>1</revision>
            <!-- The supplier is only described through its name. -->
            <org>
                <name>My Company</name>
            </org>
        </ontoml:supplier>
    </contained_suppliers>
    <!-- All the properties described in an ontology are gathered in the "contained_properties" XML element -->
    <contained_properties>
        <ontoml:property xsi:type="ontoml:NON_DEPENDENT_P_DET_Type" id="0002-38491502100024#02-
INNER_DIAMETER#001">
            <!-- Domain definition of the "inner diameter" property -->
            <name_scope class_ref="0002-38491502100024#01-BEARING#1"/>
            <!-- We assume that the provided property revision is equal to "1" -->
            <revision>1</revision>
            <!-- This property preferred name is not translated and set to "inner diameter" -->
            <preferred_name>
                <label>inner diameter</label>
            </preferred_name>
            <!-- The "inner diameter" property definition is not translated. -->
            <definition>
                <text>the bearing inner diameter</text>
            </definition>
            <!-- The "d_in" preferred symbol is assigned to this property -->
            <preferred_symbol>
                <text_representation>d_in</text_representation>
            </preferred_symbol>
            <!-- A real measure value domain (REAL_MEASURE_TYPE_Type XML complex type), expressed in
millimetre, is assigned to this property. -->
            <domain xsi:type="ontoml:REAL_MEASURE_TYPE_Type">
                <unit>
                    <structured_representation xsi:type="ontoml:SI_UNIT_Type">
                        <!-- The particular named unit used is an SI Unit, as defined by the SI_UNIT_Type XML complex
type-->
                        <prefix>MILLI</prefix>
                        <name>METRE</name>
                    </structured_representation>
                </unit>
            </domain>
        </ontoml:property>
        <ontoml:property xsi:type="ontoml:NON_DEPENDENT_P_DET_Type" id="0002-38491502100024#02-
OUTER_DIAMETER#001">
```

```
                <!-- Domain definition of the "outer diameter" property -->
                <name_scope class_ref="0002-38491502100024#01-BEARING#1"/>
                <!-- We assume that the provided property revision is equal to "1" -->
                <revision>1</revision>
                <!-- This property preferred name is not translated and set to "outer diameter" -->
                <preferred_name>
                    <label>outer diameter</label>
                </preferred_name>
                <!-- The "outer diameter" property definition is not translated. -->
                <definition>
                    <text>the bearing outer diameter</text>
                </definition>
                <!-- The "d_in" preferred symbol is assigned to this property -->
                <preferred_symbol>
                    <text_representation>d_out</text_representation>
                </preferred_symbol>
                <!-- A real measure value domain (REAL_MEASURE_TYPE_Type XML complex type), expressed in
millimetre, is assigned to this property. -->
                <domain xsi:type="ontoml:REAL_MEASURE_TYPE_Type">
                    <unit>
                        <structured_representation xsi:type="ontoml:SI_UNIT_Type">
                            <!-- The particular named unit used is an SI Unit, as defined by the SI_UNIT_Type XML complex
type-->
                            <prefix>MILLI</prefix>
                            <name>METRE</name>
                        </structured_representation>
                    </unit>
                </domain>
            </ontoml:property>
            <ontoml:property xsi:type="ontoml:NON_DEPENDENT_P_DET_Type" id="0002-38491502100024#02-
THICKNESS#001">
                <!-- Domain definition of the "thickness" property -->
                <name_scope class_ref="0002-38491502100024#01-BEARING#1"/>
                <!-- We assume that the provided property revision is equal to "1" -->
                <revision>1</revision>
                <!-- This property preferred name is not translated and set to "thickness" -->
                <preferred_name>
                    <label>thickness</label>
                </preferred_name>
                <!-- The "thickness" property definition is not translated -->
                <definition>
                    <text>the paw thickness</text>
                </definition>
                <!-- The "e" preferred symbol is assigned to this property -->
                <preferred_symbol>
                    <text_representation>e</text_representation>
                </preferred_symbol>
                <!-- A real measure value domain, expressed in millimetre (REAL_MEASURE_TYPE_Type XML complex
type), is assigned to this property. -->
                <domain xsi:type="ontoml:REAL_MEASURE_TYPE_Type">
                    <unit>
                        <structured_representation xsi:type="ontoml:SI_UNIT_Type">
                            <!-- The particular named unit used is an SI Unit, as defined by the SI_UNIT_Type XML complex
type-->
                            <prefix>MILLI</prefix>
                            <name>METRE</name>
                        </structured_representation>
                    </unit>
                </domain>
            </ontoml:property>
        </contained_properties>
        <!-- The ontology is provided in a single language, english.-->
    </dictionary>
    <!-- The "content" XML element allows to represent products according to some product charaterization classes
described in a given ontology-->
    <library xsi:type="ontoml:LIBRARY_IN_STANDARD_FORMAT_Type">
        <contained_class_extensions>
```

```
        <!-- We describe the products that relate to a product characterization class. Every product is described as a
set of (property reference, type value) pairs. -->
        <class_extension xsi:type="ontoml:EXPLICIT_ITEM_CLASS_EXTENSION_Type">
            <!-- The particular product characterization class for which the products are specified is referenced using
the "class_ref" XML attribute.-->
            <dictionary_definition class_ref="0002-38491502100024#01-PAW#1"/>
            <!-- We assume that the provided content revision is equal to "1" -->
            <content_revision>1</content_revision>
            <!-- the "instance_identification" defines which properties of the describes products play the role of a
primary key. In our example, the "inner diameter" property plays such a role. This property is therefore referenced
unambiguously using the "property_ref" XML attribute -->
            <instance_identification>
                <property property_ref="0002-38491502100024#02-INNER_DIAMETER#1"/>
            </instance_identification>
            <!-- A product characterization class consists of a set of product. they are all gathered in the "population"
XML element. -->
            <population>
                <!-- Every product is defines using external resources for describing a product as a set of (property
reference, type value) pairs.Moreover, every product references the class to which it belongs using the "class_ref"
attribute -->
                <cat:item class_ref="0002-38491502100024#01-PAW#1">
                    <!-- The first value is about the "inner diameter" property that is referenced using the "property_ref
attribute -->
                    <cat:property_value property_ref="0002-38491502100024#02-INNER_DIAMETER#1">
                        <!-- The value is a real measure, whose unit is "mm". This value is equal to "10". -->
                        <val:measure_single_number_value UOM_code="mm">
                            <val:real_value>10</val:real_value>
                        </val:measure_single_number_value>
                    </cat:property_value>
                    <!-- The second value is about the "thickness" property that is referenced using the "property_ref
attribute -->
                    <cat:property_value property_ref="0002-38491502100024#02-THICKNESS#1">
                        <!-- The value is a real measure, whose unit is "mm". This value is equal to "10". -->
                        <val:measure_single_number_value UOM_code="mm">
                            <val:real_value>1</val:real_value>
                        </val:measure_single_number_value>
                    </cat:property_value>
                    <!-- The third value is about the "outer diameter" property that is referenced using the "property_ref
attribute -->
                    <cat:property_value property_ref="0002-38491502100024#02-OUTER_DIAMETER#1">
                        <!-- The value is a real measure, whose unit is "mm". This value is equal to "10". -->
                        <val:measure_single_number_value UOM_code="mm">
                            <val:real_value>15</val:real_value>
                        </val:measure_single_number_value>
                    </cat:property_value>
                </cat:item>
                <!-- Second product description -->
                <cat:item class_ref="0002-38491502100024#01-PAW#1">
                    <cat:property_value property_ref="0002-38491502100024#02-INNER_DIAMETER#1">
                        <val:measure_single_number_value UOM_code="mm">
                            <val:real_value>11</val:real_value>
                        </val:measure_single_number_value>
                    </cat:property_value>
                    <cat:property_value property_ref="0002-38491502100024#02-THICKNESS#1">
                        <val:measure_single_number_value UOM_code="mm">
                            <val:real_value>1</val:real_value>
                        </val:measure_single_number_value>
                    </cat:property_value>
                    <cat:property_value property_ref="0002-38491502100024#02-OUTER_DIAMETER#1">
                        <val:measure_single_number_value UOM_code="mm">
                            <val:real_value>16.5</val:real_value>
                        </val:measure_single_number_value>
                    </cat:property_value>
                </cat:item>
                <!-- Third product description -->
                <cat:item class_ref="0002-38491502100024#01-PAW#1">
                    <cat:property_value property_ref="0002-38491502100024#02-INNER_DIAMETER#1">
```

```
                        <val:measure_single_number_value UOM_code="mm">
                            <val:real_value>13</val:real_value>
                        </val:measure_single_number_value>
                    </cat:property_value>
                    <cat:property_value property_ref="0002-38491502100024#02-THICKNESS#1">
                        <val:measure_single_number_value UOM_code="mm">
                            <val:real_value>2</val:real_value>
                        </val:measure_single_number_value>
                    </cat:property_value>
                    <cat:property_value property_ref="0002-38491502100024#02-OUTER_DIAMETER#1">
                        <val:measure_single_number_value UOM_code="mm">
                            <val:real_value>19.5</val:real_value>
                        </val:measure_single_number_value>
                    </cat:property_value>
                </cat:item>
                <!-- Fourth product description -->
                <cat:item class_ref="0002-38491502100024#01-PAW#001">
                    <cat:property_value property_ref="0002-38491502100024#02-INNER_DIAMETER#001">
                        <val:measure_single_number_value UOM_code="mm">
                            <val:real_value>17</val:real_value>
                        </val:measure_single_number_value>
                    </cat:property_value>
                    <cat:property_value property_ref="0002-38491502100024#02-THICKNESS#001">
                        <val:measure_single_number_value UOM_code="mm">
                            <val:real_value>3</val:real_value>
                        </val:measure_single_number_value>
                    </cat:property_value>
                    <cat:property_value property_ref="0002-38491502100024#02-OUTER_DIAMETER#001">
                        <val:measure_single_number_value UOM_code="mm">
                            <val:real_value>25.5</val:real_value>
                        </val:measure_single_number_value>
                    </cat:property_value>
                </cat:item>
                <!-- Fifth product description -->
                <cat:item class_ref="0002-38491502100024#01-PAW#001">
                    <cat:property_value property_ref="0002-38491502100024#02-INNER_DIAMETER#001">
                        <val:measure_single_number_value UOM_code="mm">
                            <val:real_value>19</val:real_value>
                        </val:measure_single_number_value>
                    </cat:property_value>
                    <cat:property_value property_ref="0002-38491502100024#02-THICKNESS#001">
                        <val:measure_single_number_value UOM_code="mm">
                            <val:real_value>4</val:real_value>
                        </val:measure_single_number_value>
                    </cat:property_value>
                    <cat:property_value property_ref="0002-38491502100024#02-OUTER_DIAMETER#001">
                        <val:measure_single_number_value UOM_code="mm">
                            <val:real_value>28.5</val:real_value>
                        </val:measure_single_number_value>
                    </cat:property_value>
                </cat:item>
            </population>
            <table_like>true</table_like>
        </class_extension>
    </contained_class_extensions>
    <responsible_supplier supplier_ref="0002-38491502100024"/>
  </library>
</ontoml:ontoml>
```

Приложение J

(справочное)

Информация для поддержки внедрения стандарта

Для оказания помощи по внедрению настоящего стандарта может предоставляться дополнительная информация, которую можно найти по адресу (URL):

http://www.tc184-sc4.org/implementation information/13584/00032

Приложение ДА

(справочное)

Сведения о соответствии ссылочных международных стандартов ссылочным национальным стандартам Российской Федерации

Таблица ДА.1

Обозначение ссылочного международного стандарта	Степень соответствия	Обозначение и наименование соответствующего национального стандарта
ИСО 10303-11:2009	IDT	ГОСТ Р ИСО 10303-11:2009 «Системы автоматизации производства и их интеграция. Представление данных об изделии и обмен этими данными. Часть 11. Методы описания. Справочное руководство по языку EXPRESS»
ИСО/МЭК 14977:1996		*
ИСО/ТС 29002-5:2009		*
ИСО/ТС 29002-10:2009		*

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

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

IDT – идентичный стандарт.

Библиография

[1] ИСО 639-1:2002 (ISO 639-1:2002) — Коды для представления названий языков. Часть 1. Двухбуквенный код (Codes for the representation of names of languages. Part 1. Alpha-2 code)

[2] ИСО 639-2:2002 (ISO 639-2:2002) — Коды для представления названий языков. Часть 2. Трехбуквенный код (Codes for the representation of names of languages - Part 2: Alpha-3 code)

[3] ИСО 843:1997 (ISO 843:1997) — Информация и документация. Транслитерация и транскрипция букв греческого алфавита буквами латинского алфавита (Information and documentation - Conversion of Greek characters into Latin characters)

[4] ИСО 3166-1:2006 (ISO 3166-1:2006) — Коды для представления названий стран и единиц их административно-территориального деления. Часть 1. Коды стран (Codes for the representation of names of countries and their subdivisions - Part 1: Country codes)

[5] ИСО 10303-1:1994 (ISO 10303-1:1994) — Системы промышленной автоматизации и интеграция. Представление данных о продукции и обмен данными. Часть 1. Обзор и основные принципы (Industrial automation systems and integration - Product data representation and exchange - Part 1: Overview and fundamental principles)

[6] ИСО 10303-21:2002 (ISO 10303-21:2002) — Системы промышленной автоматизации и интеграция. Представление данных о продукции и обмен данными. Часть 21. Методы реализации. Кодирование открытого текста структуры обмена (Industrial automation systems and integration - Product data representation and exchange - Part 21: Implementation methods: Clear text encoding of the exchange structure)

[7] ИСО 10303-41:2005 (ISO 10303-41:2005) — Системы промышленной автоматизации и интеграция. Представление данных о продукции и обмен данными. Часть 41. Интегрированные родовые ресурсы. Основы описания продукции и программного обеспечения (Industrial automation systems and integration - Product data representation and exchange - Part 41: Integrated generic resource: Fundamentals of product description and support)

[8] ИСО 10303-42:2003 (ISO 10303-42:2003) — Системы промышленной автоматизации и интеграция. Представление данных о продукции и обмен данными. Часть 42. Интегрированные родовые ресурсы. Геометрическое и топологическое представление (Industrial automation systems and integration - Product data representation and exchange - Part 42: Integrated generic resource: Geometric and topological representation)

[9] ИСО 13584-1:2001 (ISO 13584-1:2001) — Системы промышленной автоматизации и интеграция. Библиотека данных на детали. Часть 1. Обзор и основные принципы (Industrial automation systems and integration. Parts library. Part 1. Overview and fundamental principles)

[10] ИСО 13584-24:2003 (ISO 13584-24:2003) — Системы промышленной автоматизации и интеграция. Библиотека данных на детали. Часть 24. Логический ресурс. Логическая модель библиотеки поставщика (Industrial automation systems and integration - Parts library - Part 24: Logical resource: Logical model of supplier library)

[11] ИСО 13584-25:2004 (ISO 13584-25:2004) — Системы промышленной автоматизации и интеграция. Библиотека данных на детали. Часть 25. Логический ресурс: логическая модель библиотеки поставщика с агрегированными значениями и подробным содержанием (Industrial automation systems and integration - Parts library - Part 25: Logical resource: Logical model of supplier library with aggregate values and explicit content)

[12] ИСО 13584-26:2000	Системы промышленной автоматизации и интеграция. Библиотека данных на детали. Часть 26. Логический ресурс: идентификация поставщика информации
(ISO 13584-26:2000)	(Industrial automation systems and integration - Parts library - Part 26: Logical resource: Information supplier identification)
[13] ИСО 13584-42:2010	Системы промышленной автоматизации и интеграция. Библиотека данных на детали. Часть 42. Методология описания: методология структурирования групп деталей
(ISO 13584-42:2010)	(Industrial automation systems and integration - Parts library - Part 42: Description methodology: Methodology for structuring parts families)
[14] ИСО 13584-102:2006	Системы промышленной автоматизации и интеграция. Библиотека данных на детали. Часть 102. Протокол обмена представлениями по спецификации соответствия ИСО 10303
(ISO 13584-102:2006)	(Industrial automation systems and integration - Parts library - Part 102: View exchange protocol by ISO 10303 conforming specification)
[15] ИСО 13584-501:2007	Системы промышленной автоматизации и интеграция. Библиотека данных на детали. Часть 501. Справочный словарь измерительных инструментов. Процедура регистрации
(ISO 13584-501:2007)	(Industrial automation systems and integration - Parts library - Part 501: Reference dictionary for measuring instruments - Registration procedure)
[16] ИСО 13584-511:2006	Системы промышленной автоматизации и интеграция. Библиотека данных на детали. Часть 511. Механические системы и компоненты общего назначения. Справочный словарь по крепежу
(ISO 13584-511:2006)	(Industrial automation systems and integration - Parts library - Part 511: Mechanical systems and components for general use - Reference dictionary for fasteners)
[17] ИСО/ТС 23768-1:2011	Роликовые подшипники. Библиотека деталей. Часть 1. Справочный словарь
(ISO/TS 23768-1:2011)	(Rolling bearings - Parts library - Part 1: Reference dictionary for rolling bearings)
[18] ИСО/ТС 29002-20:2010	Промышленные автоматические системы и интеграция. Обмен характеристическими данными. Часть 20. Услуги по идентификации концептуального словаря
(ISO/TS 29002-20:2010)	(Industrial automation systems and integration -Exchange of characteristic data - Part 20: Concept dictionary resolution services)
[19] ИСО 80000-1:2009	Величины и единицы. Часть 1. Общие положения
(ISO 80000-1:2009)	(Quantities and units -- Part 1: General)
[20] МЭК 61360 (все части)	Стандартные типы элементов данных с соответствующей схемой классификации для электрических компонентов
(IEC 61360 (all parts))	(Standard data elements types with associated classification scheme for electric items)
[21] Руководство ИСО/МЭК 77-2:2008	Руководство для спецификации свойств и классов изделия. Часть 2. Технические принципы и руководство
(ISO/IEC Guide 77-2:2008)	(Guide for specification of product properties and classes -- Part 2: Technical principles and guidance)
[22] ИСО/МЭК 8824-1:2008	Информационные технологии. Нотация абстрактного синтаксиса версии 1 (ASN.1). Часть 1. Спецификация базовой нотации
(ISO/IEC Guide 8824-1:2008)	(Information technology -- Abstract Syntax Notation One (ASN.1): Specification of basic notation)

УДК 658.52.011.56:006.354 ОКС 25.040.40

Ключевые слова: автоматизированные промышленные системы, интеграция, жизненный цикл систем, управление производством

Электронное издание

Тираж 14 экз. Зак. 2Е.

Подготовлено на основе электронной версии, предоставленной разработчиком стандарта

ФГУП «СТАНДАРТИНФОРМ»

123995 Москва, Гранатный пер., 4.

www.gostinfo.ru info@gostinfo.ru