ФЕДЕРАЛЬНОЕ АГЕНТСТВО

ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р ИСО 13606-2 — 2012

Информатизация здоровья

ПЕРЕДАЧА ЭЛЕКТРОННЫХ МЕДИЦИНСКИХ КАРТ

Часть 2

Спецификация передачи архетипов

ISO 13606-2:2008
Health informatics — Electronic health record communication — Part 2: Archetype interchange specification
(IDT)

Издание официальное

Москва
Стандартинформ
2014

gost.e-ecolog.ru

Предисловие

1 ПОДГОТОВЛЕН Федеральным государственным бюджетным учреждением «Центральный научно-исследовательский институт организации и информатизации здравоохранения Министерства здравоохранения и социального развития Российской Федерации» (ФГБУ ЦНИИОИЗ Минздравсоцразвития РФ) и Федеральным государственным автономным научным учреждением «Центральный научно-исследовательский и опытно-конструкторский институт робототехники и технической кибернетики» на основе собственного аутентичного перевода на русский язык стандарта, указанного в пункте 4

2 ВНЕСЕН Техническим комитетом по стандартизации ТК 468 «Информатизация здоровья» при ФГБУ ЦНИИОИЗ Минздравсоцразвития РФ — единоличным представителем ИСО/ТК 215

3 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Приказом Федерального агентства по техническому регулированию и метрологии от 29 октября 2012 г. № 584-ст

4 Настоящий стандарт идентичен международному стандарту ИСО 13606-2:2008 «Информатизация здоровья. Передача электронных медицинских карт. Часть 2. Спецификация передачи архетипов» (ISO 13606-2:2008 «Health informatics — Electronic health record communication — Part 2: Archetype interchange specification»).

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)

© Стандартинформ, 2014

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

Содержание

1 Область применения . . . 1
2 Соответствие . . . 1
3 Нормативные ссылки . . . 1
4 Термины и определения . . . 2
5 Сокращения . . . 6
6 Требования к представлению архетипов . . . 3
6.1 Общие сведения . . . 3
6.2 Определение, описание архетипа и информация о его публикации . . . 3
6.3 Ограничения узлов архетипа . . . 5
6.4 Ограничения значений данных . . . 7
6.5 Профиль по отношению к базовой модели EN 13606-1 . . . 8
7 Модель архетипов . . . 10
7.1 Введение . . . 10
7.2 Обзор . . . 11
7.3 Пакет archetype . . . 15
7.4 Пакет archetype_description . . . 17
7.5 Пакет constraint_model . . . 21
7.6 Пакет assertion . . . 27
7.7 Пакет primitive . . . 30
7.8 Пакет ontology . . . 36
7.9 Пакет domain_extensions . . . 38
7.10 Пакет support . . . 41
7.11 Пакет generic_types . . . 48
7.12 Расширения, специфичные для предметной области (справочно) . . . 49
8 Язык определения архетипов (ADL) . . . 50
8.1 Язык dADL — ADL данных . . . 50
8.2 Язык cADL — ADL ограничений . . . 64
8.3 Утверждения . . . 84
8.4 Пути в языке ADL . . . 86
8.5 ADL — язык определения архетипов . . . 87
Приложение ДА (справочное) Сведения о соответствии ссылочных международных стандартов ссылочным национальным стандартам Российской Федерации . . . 97
Библиография . . . 98

Введение

Комплекс стандартов ИСО 13606 состоит из следующих частей под общим заголовком «Информатизация здоровья. Передача электронных медицинских карт»:

часть 1: «Базовая модель»;

часть 2: «Спецификация передачи архетипов»;

часть 3: «Базовые архетипы и списки терминов»;

часть 4: «Безопасность»;

часть 5: «Спецификация интерфейсов».

Стандарт ИСО 13606-2 подготовлен техническим комитетом ИСО/ТК 215 «Информатизация здоровья».

Ведение электронных медицинских карт, предназначенных для использования во многих учреждениях в течение длительного времени, на практике может быть обеспечено с помощью объединения медицинских приложений, баз данных (и все чаще медицинских приборов), которые хорошо приспособлены к конкретным условиям, специальностям или учреждениям.

При этом требуется, чтобы данные, поступившие в электронную медицинскую карту (ЭМК) из разных систем, могли отображаться на единое всестороннее представление, допускающее обратное преобразование и предназначенное для упрощения разработки интерфейсов и сообщений в рамках распределенной сети (федерации) систем и сервисов ЭМК. Такое представление должно быть достаточно общим и обладать широкими возможностями, позволяющими представлять данные медицинской карты, составляющие часть или всю ЭМК (или несколько ЭМК) и являющиеся предметом передачи.

Подход, принятый в международных стандартах комплекса ИСО 13606 и подкрепленный международными исследованиями в области систем ведения ЭМК, должен был определить строгую обобщенную базовую модель, позволяющую такое описание всех видов данных и структур данных в ЭМК, при котором любая разметка и любой контекст являются неотъемлемой частью содержания. Выписка из ЭМК (в соответствии с определением из ИСО 13606-1) должна содержать все имена, структуру и контекст, необходимые для ее точной интерпретации принимающей стороной, даже если ее построение и характер медицинского содержания не были «согласованы» заранее.

Однако широкое совместное использование медицинских карт и анализ их содержания в различных пунктах требуют также согласованного подхода к клиническим (семантическим) структурам данных, передаваемым в соответствии с базовой моделью так, чтобы эквивалентная информация представлялась в согласованной форме. Это необходимо для обеспечения безопасной обработки данных ЭМК, поступивших из разных источников.

Архетипы

Таким образом, сложность обеспечения интероперабельности систем ведения ЭМК состоит в необходимости разработки обобщенного подхода для единообразного представления всевозможных видов структур электронных медицинских карт. Такой подход должен позволить обрабатывать записи ЭМК, поступающие от специалистов разных профессий, специализаций или служб, с учетом того, что совокупности клинических данных, наборы значений, шаблоны и т. п., используемые в разных областях здравоохранения, отличаются разнообразием, сложностью и будут часто изменяться по мере развития клинической практики и медицинских знаний. Данное требование является частью хорошо известной проблемы семантической интероперабельности в информатизации здоровья.

В подходе, принятом в настоящем стандарте, различаются «базовая модель» (Reference Model), которая используется для представления общих свойств информации, содержащейся в медицинских картах, и «архетипы» (Archetypes) (соответствующие «модели архетипов»), являющиеся метаданными, применяемыми для определения шаблонов представления специфических характеристик медицинских данных, отражающих требования каждой конкретной профессии, специальности или службы.

Базовая модель определена в соответствии с информационной точкой зрения в терминах модели открытой распределенной обработки (ОРО). Она описывает представление глобальных характеристик компонентов медицинских карт, способы их агрегирования, а также контекстную информацию, необходимую для удовлетворения этических, юридических и местных требований. Базовая модель определена в стандарте ИСО 13606-1, являющемся частью комплекса международных стандартов ИСО 13606. Эта модель определяет совокупность классов, образующих типовые строительные блоки ЭМК. Они отражают устойчи-

вые характеристики электронной медицинской карты и должны использоваться в распределенной (федеративной) системе ведения ЭМК в форме конкретных сообщений или интерфейсов (как это определено в стандарте ИСО 13606-5).

Архетипы являются эффективными, заранее согласованными сочетаниями именованных иерархий классов RECORD_COMPONENT, принятыми сообществом пользователей для обеспечения семантической интероперабельности, непротиворечивости и качества данных.

Для класса EHR_EXTRACT, определенного в ИСО 13606-1, архетип определяет (и эффективно ограничивает) конкретную иерархию подклассов класса RECORD_COMPONENT, определяя и ограничивая их имена и релевантные значения других атрибутов, обязательность и кратность каждого узла иерархии, типы данных и диапазоны значений, которые могут принимать значения данных класса ELEMENT. Архетип может содержать также другие ограничения зависимостей. Экземпляры архетипа, в свою очередь, соответствуют формальной модели, называемой «моделью архетипов» (которая является моделью ограничений, описывающей информационную точку зрения в терминах модели ОРО). Хотя модель архетипов стабильна, отдельные экземпляры архетипов могут пересматриваться или заменяться другими по мере развития клинической практики. Чтобы новые редакции архетипа не сделали недействительными данные, созданные в предыдущих редакциях, применяется контроль версий.

Архетипы могут использоваться в системах ведения ЭМК для управления данными ЭМК, отправленными в хранилища. Однако настоящий стандарт описывает требования к передаче данных ЭМК и поэтому в нем не делается никаких предположений о наличии архетипов в системе поставщика ЭМК. Предполагается, что исходные данные ЭМК системы, не содержащей архетипы, при необходимости могут быть отображены на совокупность архетипов при создании выписки EHR_EXTRACT.

Базовая модель, определенная в ИСО 13606-1, описывает атрибуты, применяемые для указания архетипа, которому соответствует любой объект класса RECORD_ COMPONENT, содержащийся в выписке EHR_EXTRACT. Класс RECORD_COMPONENT содержит атрибут *archetype_id*, предназначенный для идентификации архетипа и узла, которым соответствует этот класс. В случае данных, для представления которых использованы архетипы, атрибут *meaning* относится к исходному понятию, с которым связан соответствующий узел архетипа. Однако необходимо отметить, что ИСО 13606-1 не требует участия архетипов в управлении иерархией объектов RECORD_COMPONENT в выписке EHR_EXTRACT; в данной модели атрибуты, относящиеся к архетипам, необязательны. Тем самым учтено, что международное принятие подхода на основе архетипов будет постепенным в течение ряда лет.

Фонды архетипов

Число архетипов, необходимых для совместного использования ЭМК медицинским сообществом, будет зависеть от разнообразия его деятельности. Полный набор архетипов, необходимых для целой страны, неизвестен, но со временем в глобальном масштабе число архетипов может достичь нескольких тысяч. Идеальной основой знаний для разработки определений архетипов будут клинические методические указания, стандарты ведения пациентов, научные публикации и другие сведения о передовом опыте. Однако де-факто к источникам согласованных структур клинических данных могут также относиться:

- схемы данных (модели) существующих медицинских информационных систем;
- компьютерные экранные формы, используемые в таких системах для ввода данных и отображения результатов анализа;
- шаблоны ввода данных, всплывающие списки и справочники, используемые в таких системах;
- наборы медицинских данных коллективного доступа, сообщения и отчеты, используемые в местном или в национальном масштабе;
- структура форм, используемых для документирования медицинских консультаций или заключений в бумажных медицинских картах;
- медицинская информация, используемая в совокупностях вторичных данных;
- заранее согласованные термины в терминологических системах.

Хотя этот список фактически применяемых способов представления структур клинических данных достаточно устоялся, используемые в них форматы представления данных очень редко являются интероперабельными. Стандартизованные архетипы обеспечивают интероперабельный способ представления и совместного применения этих спецификаций для обеспечения качественного ведения медицинских карт и семантической интероперабельности совместно используемых ЭМК.

Привлечение национальных служб здравоохранения, академических организаций и профессиональных органов к разработке архетипов даст возможность использовать данный подход для достижения надежности медицинской практики, основанной на принципах доказательной медицины. Следующей ключевой задачей является стимулирование медицинских сообществ к созданию библиотек архетипов. Определение хода работы находится вне области применения настоящего стандарта, но национальные программы здравоохранения, по крайней мере, в нескольких странах начинают стимулировать коллективы разработчиков медицинских информационных технологий к разработке и вводу в действие совокупностей архетипов, удовлетворяющих потребностям конкретных областей здравоохранения. Будут создаваться региональные или национальные общедоступные библиотеки определений архетипов с доступом через Интернет, которые могут использоваться в системах ведения ЭМК местного уровня. Для этого потребуются процессы верификации и сертификации качества совместно используемых архетипов, что также выходит за рамки настоящего стандарта, но продвигается некоммерческими организациями, например openEHR Foundation и EuroRec Institute.

Передача архетипов

Настоящий стандарт определяет требования к полному и интероперабельному представлению архетипов и определяет информационную точку зрения на модель архетипов в терминах ОРО, а также необязательный формат передачи архетипов, называемый «языком определения архетипов» (ADL).

Настоящий стандарт не требует, чтобы в качестве внутренней архитектуры фондов, сервисов или компонентов архетипов, применяемых для создания, хранения и использования архетипов совместно со службами ведения ЭМК, была принята какая-то конкретная модель. Однако он требует, чтобы такие архетипы могли отображаться на определенную в настоящем стандарте модель архетипов с целью обеспечения передачи и интероперабельности ЭМК в рамках сообщества, совместно использующего ЭМК.

Обзор модели архетипов

Ниже приведено общее справочное описание модели, определенной в разделе 7.

Полная модель архетипа состоит из идентифицирующей информации, описания (метаданных архетипа), определения (выраженного в терминах ограничений экземпляров объектной модели) и онтологии. Идентифицирующая информация и состояние жизненного цикла являются частью класса *ARCHETYPE*. Описание архетипа разделено на историю версий и описательную информацию об архетипе. История версий связана с передачей архетипа в фонд и имеет форму списка, содержащего записи регистрационного журнала версий, а описательная информация отражает собственно архетип (независимо от того, был ли он помещен в фонд какого-либо вида).

«Главная» часть модели архетипов, а именно, определение архетипа, представляет собой экземпляр объекта *C_COMPLEX_OBJECT*, так как корень структуры ограничений архетипа всегда должен принимать форму ограничения объектного типа данных, не являющегося примитивным. Четвертая основная часть модели архетипа – онтология представляется своим собственным классом, что позволяет архетипам быть нейтральными по отношению к естественному языку и терминологии.

В пакет архетипа включен также перечисляемый тип данных *VALIDITY_KIND*. Он предназначен для использования в качестве типа данных любого атрибута в модели ограничений, имеющего логические значения «обязательный», «необязательный» или «запрещенный».Он используется в данной модели в классах *C_Date*, *C_Time* и *C_Date_Time*.

Архетипы содержат некоторые элементы естественного языка, включая описание и определения онтологии. Поэтому каждый архетип создается на некотором исходном языке, который указывается в атрибуте *original_language* класса *ARCHETYPE*. Перевод архетипа осуществляется следующим образом:

- на новый язык переводится каждый элемент, зависящий от языка;
- к объекту *ARCHETYPE.translations* добавляется новый экземпляр класса *TRANSLATION_DETAILS*, содержащий подробные сведения о переводчике, организации, контроле качества и т. д.

Функция *languages_available* дает полный список языков архетипа.

Определение архетипа

Основная часть модели архетипа состоит из чередующихся слоев узлов, ограничивающих объекты и атрибуты, каждый из которых содержит следующий уровень узлов. В настоящем подразделе под термином «атрибут» понимается любое свойство данных некоторого класса, независимо от того, является ли оно атрибутом «отношения» (т. е. ассоциацией, агрегированием или композицией) или «примитивным» атрибутом (т.е. значением). Конечными элементами графа являются узлы ограничителей примитивных объектов,

ограничивающие примитивные типы данных, например String, Integer и т. д. Существуют также узлы, представляющие собой внутренние ссылки на другие узлы, узлы ссылок на ограничения, ссылающиеся на текстовые ограничения других архетипов, которые могут появиться в данной точке.

Полный список типов узлов выглядит следующим образом:

- C_complex_object — любой внутренний узел, представляющий ограничение экземпляров непримитивного типа данных, например ENTRY, SECTION;
- C_attribute — узел, представляющий ограничение атрибута объектного типа данных (т. е. в терминах языка UML — «отношение» или «примитивный атрибут»);
- C_primitive_object — узел, представляющий ограничение примитивного (встроенного) объектного типа данных;
- Archetype_internal_ref — узел, ссылающийся на ранее определенный объектный узел того же архетипа, ссылка делается с использованием пути;
- Constraint_ref — узел, который (обычно) ссылается на ограничение текстового объекта или объекта, имеющего кодированные значения. Он появляется в разделе онтологии архетипа и в языке ADL и обозначается с помощью кода «acNNNN»; ограничение выражается в терминах запроса к внешнему объекту, обычно к терминологии или онтологии;
- Archetype_slot – узел, формулировки которого задают ограничение, определяющее другие архетипы, возможные в данной точке рассматриваемого архетипа; логически у данного узла та же семантика, что и у узла C_COMPLEX_OBJECT за исключением того, что ограничения выражены для другого архетипа, а не для рассматриваемого.

Описание архетипа

Все, что обычно считается «метаданными» архетипа, т. е. его создатель, дата создания, назначение и другие описательные элементы, представляется с помощью классов *ARCHETYPE_DESCRIPTION* и *ARCHETYPE_DESCRIPTION_ITEM*. Части описания, выраженные на естественном языке, для которых могут потребоваться версии на других языках, представляются экземплярами класса *ARCHETYPE_ DESCRIPTION_ITEM*. Если в объекте *ARCHETYPE_DESCRIPTION* содержится более одного объекта *ARCHETYPE_DESCRIPTION_ITEM*, то каждый из них должен содержать в точности ту же самую информацию на разных естественных языках.

Если архетип переводится для использования в другой языковой среде, то каждый объект *ARCHETYPE_DESCRIPTION_ITEM* должен быть скопирован и переведен на новый язык.

Класс *AUDIT_DETAILS* связан с помещением архетипа в фонд и контролем его версий. В реальном архетипе каждый экземпляр этого класса описывает одно действие по помещению версии архетипа в фонд и содержит атрибуты, документирующие кто, когда и почему выполнил данное действие.

Примечание — Версия архетипа должна ограничиваться модификацией описательной информации и добавлением переводов на другой язык и/или связи терминов. Если часть, содержащая определение архетипа, перестает быть действительной, то данный архетип должен быть заменен новым архетипом, чтобы обеспечить согласование соответствующих экземпляров данных ЭМК со спецификацией того же архетипа.

Онтология архетипа

Все лингвистические объекты определяются в разделе онтологии архетипа. В онтологии архетипа имеются четыре основные части: определения терминов, определения ограничений, связи терминов и связи ограничений. Первые две части описывают смысл разных терминов и ограничений, выраженных в текстовой форме, которые встречаются в архетипе; им присваиваются уникальные идентификаторы, используемые в теле определения а рхетипа. Последние две части онтологии описывают отображения внутренне применяемых терминов на внешние терминологии.

Специализация архетипа

Архетипы могут быть специализированы: архетип считается специализацией другого архетипа, если он определяет этот архетип как своего «родителя», и вносит в его определение только такие изменения, которые накладывают на него «более строгие» ограничения, чем ограничения родителя. Любые данные, созданные с помощью использования специализированного архетипа, должны быть совместимы как с ним, так и с его родителем.

Каждый архетип имеет «глубину специализации». У архетипов, не имеющих родителя, глубина равна 0, а специализированные архетипы добавляют один уровень к своей глубине на каждом шаге вниз по иерархии, необходимом для их достижения.

Композиция архетипов

Для образования более крупных структур, семантически эквивалентных одному большому архетипу, могут использоваться композиции архетипов. Средствами композиции являются слоты архетипов, которые сами определяются в терминах ограничений.

Типы данных и пакет поддержки

Модель зависит от трех групп допустимых типов данных, чьи имена и допустимые семантики описаны в ИСО/МЭК 11404.

Первая группа содержит наиболее общие типы данных:

- Any (любой);
- Boolean (булевский);
- Character (символьный);
- Integer (целый);
- Real (действительный);
- Double (двойной точности);
- String (строковый).

Вторая группа содержит допустимые библиотечные типы данных:

- date (дата);
- time (время);
- date_time (дата и время);
- duration (длительность).

Эти типы данных поддерживаются в большинстве технологий реализации, включая языки XML, Java и другие языки программирования. В настоящем стандарте они не определяются, что позволяет отображать их на наиболее подходящие конкретные типы данных в каждой технологии реализации.

Третья группа содержит обобщенные типы данных:

- List<T> (упорядоченный список, дубликаты допустимы);
- Set<T> (неупорядоченный список , дубликаты недопустимы);
- Hash <T, K> (список односторонних сверток элементов любого типа);
- Interval <T> (интервал экземпляров любого упорядоченного типа).

Хотя указанные типы данных поддерживаются в большинстве технологий реализации, они пока что не представлены в языке UML. Поэтому семантика этих типов данных определена в пакете Generic_Types модели на языке UML.

Остальные необходимые типы данных определены в пакете поддержки (Support Package) модели архетипа:

- ARCHETYPE_ID (идентификатор архетипа);
- HIER_OBJECT_ID (идентификатор иерархии объектов);
- TERMINOLOGY_ID (идентификатор терминологии);
- CODE_PHRASE (кодированное предложение);
- CODED_TEXT (кодированный текст).

Пакет поддержки также содержит два перечисляемых класса, предоставляющие контролируемые наборы данных, необходимые в настоящем стандарте.

Пакет ограничений модели

Любое определение архетипа представляет собой экземпляр класса *C_COMPLEX_OBJECT*, который описывает ограничения класса в основополагающей базовой модели (см. ИСО 13606-1), записанные в атрибуте *rm_type_name*. Класс *C_COMPLEX_OBJECT* содержит атрибуты типа *C_ATTRIBUTE*, являющиеся ограничениями атрибутов (т. е. любого свойства, включая отношения) данного класса базовой модели. Соответственно, каждый экземпляр класса *C_ATTRIBUTE* содержит имя ограничиваемого атрибута (в атрибуте *rm_attnibute_name*), существование и кратность, заданные ограничением (в зависимости от того, является ли ограничиваемый атрибут кратным или одиночным отношением), а также ограничение объекта, на который ссылается данный экземпляр класса *C_ATTRIBUTE* с помощью своего атрибута *children* (в соответствии с его базовой моделью) в форме новых объектов класса *C_OBJECT*.

Основными подтипами класса *C_OBJECT* являются:

- *C_COMPLEX_OBJECT (комплексный объект)*;
- *C_PRIMITIVE_OBJECT (примитивный объект)*;
- *ARCHETYPE_SLOT (слот архетипа)*.

Классы *ARCHETYPE_INTERNAL_REF* и *CONSTRAINT_REF* используются для описания, соответственно, «слота», в котором будущие архетипы могут быть использованы для продолжения описания ограничений, ссылки на часть данного архетипа, которая описывает те же самые ограничения, необходимые в другом месте, а также ссылки на ограничение ограничения, определенного в онтологии архетипа, которое, в свою очередь, указывает на внешний источник знаний, например на терминологию.

Результатом данной модели является создание структур описания архетипов, представляющих собой иерархическое чередование ограничений объектов и атрибутов. Иерархическая структура архетипа в форме повторяющихся пар объект/атрибут обеспечивает основу для использования путей, позволяющих ссылаться на любой узел архетипа. Пути в архетипе подчиняются синтаксису, который является подмножеством синтаксиса W3C Xpath.

Все типы узлов

Все узлы в структуре ограничений архетипа являются экземплярами супертипа *ARCHETYPE_CONSTRAINT*, который определяет ряд важных свойств, общих для всех узлов.

Булевское значение атрибута *any_allowed* «истина» означает, что любое значение, разрешенное базовой моделью для данного атрибута, допускается и архетипом; его использование обеспечивает простое выражение логической идеи полностью «открытого» ограничения, не требующего формирования какой-либо дальнейшей субструктуры.

Узлы атрибутов

Ограничения атрибутов описываются экземплярами двух подтипов класса *C_ATTRIBUTE*, а именно, *C_SINGLE_ATTRIBUTE* и *C_MULTIPLE_ATTRIBUTE*. Для обоих подтипов общее ограничение состоит в возможности существования соответствующего экземпляра (заданного атрибутом *rm_attribute_name*). Оба подтипа имеют список потомков, описывающих ограничения значений объекта (или объектов) атрибута.

Атрибуты с единственным значением ограничены экземпляром типа данных *C_SINGLE_ATTRIBUTE*, который использует потомков для представления множества альтернативных объектных ограничений значения атрибута.

Атрибуты с множественными значениями ограничены экземпляром типа данных *C_MULTIPLE_ATTRIBUTE*, который допускает сосуществование множества объектов, образующих контейнер, представляющий значение ограничиваемого атрибута, наряду с ограничением кратности, описывающего упорядочение и уникальность содержания данного контейнера.

Параметр *cardinality* (кратность) требуется только для таких контейнеров объектов, как *List<T>*, *Set<T>*, *BAG<T>* и т. д., тогда как параметр *existence* (существование) требуется всегда. Если используются оба параметра, то это имеет следующий смысл: ограничение существования указывает, будет ли данный объект находиться в контейнере (вообще), тогда как ограничение кратности указывает, сколько элементов должно находиться в контейнере, и организован ли он логически как список (list), множество (set) или пакет (bag).

Примитивные типы данных

Ограничения примитивных типов данных описываются классами, наследуемыми от класса *C_PRIMITIVE*, а именно, *C_STRING*, *C_INTEGER* и т. д.

Ссылки на ограничения

Класс *CONSTRAINT_REF* является представителем множества ограничений объекта, которые обычно должны быть представлены в конкретной точке в архетипе в виде объекта *C_COMPLEX_OBJECT*, но в котором действующее определение ограничения выражается в форме связи с запросом или выражением, относящимся к внешней службе (онтологической или терминологической службе), например:

- набор допустимых объектов CODED_TERM, например типов гепатита;
- выражение INTERVAL<QUANTITY>, определяющее базовый диапазон;
- множество единиц или свойств, или других числовых элементов.

Утверждения

Класс *C_ATTRIBUTE* и подтипы класса *C_OBJECT* дают возможность описывать ограничения в структурированной форме. Кроме того, любой экземпляр класса *C_COMPLEX_OBJECT* может включать в себя один или несколько *инвариантов*. Инвариантами являются выражения, представленные в форме логики предикатов, которые могут использоваться для установления ограничений частей объекта. Они не требуются для задания ограничений единственного атрибута (так как это можно сделать с помощью подходящего объекта *C_ATTRIBUTE*), но они необходимы для задания ограничений нескольких атрибутов, например ограничения, что «систолическое давление должно быть ≥ диастолического давления» в архетипе понятия «измерение кровяного давления». Подобные инварианты могут быть выражены с использованием синтаксиса, производного от синтаксиса языка OCL, разработанного в Object Management Group (OMG).

Утверждения также используются в объектах *ARCHETYPE_SLOT*, чтобы указать для слота «включенные» и «исключенные» архетипы.

Утверждения моделируются в форме дерева обобщенных выражений, состоящих из унарных префиксных (например *not* p) и бинарных инфиксных (например p *and* q) операторов.

Атрибут node_id и пути

Атрибут *node_id* в классе *C_OBJECT* и всех его подтипах выполняет две функции:

- позволяет индивидуально идентифицировать узлы ограничений объектов архетипа и, в частности, гарантирует уникальную идентификацию сестринского узла;
- является основной связью между определением архетипа (т. е. ограничениями) и онтологией архетипа, поскольку каждый атрибут *node_id* является «кодом термина» в онтологии.

Наличие в архетипе атрибутов *node_id* позволяет создавать в нем пути, указывающие каждый узел.

Расширения, специфичные для предметной области

Главная часть модели ограничений архетипа позволяет в соответствии со стандартом архетипировать (т. е. ограничивать) любой тип данных базовой модели с помощью регулярной последовательности объектов: *C_COMPLEX_OBJECT* / *C_ATTRIBUTE* / *C_PRIMITIVE_OBJECT*. Однако для описания типов данных, соответствующих логическим «листьям» более низких уровней, может потребоваться специальная семантика ограничений, которая обычно не используется при стандартном подходе. Чтобы обеспечить интеграцию таких классов в общую модель ограничений, вводится класс *C_DOMAIN_TYPE*. Это дает возможность создавать особые классы «*C_*», унаследованные от *C_DOMAIN_TYPE*, которые описывают пользовательскую семантику специфических типов данных базовой модели. Например, может быть создан класс с именем *C_QUANTITY*, имеющий семантику ограничений, отличную от используемой по умолчанию семантики последовательности *C_COMPLEX_OBJECT* / *C_ATTRIBUTE*, представляющей эти ограничения обычным образом (т. е. на основе базовой модели).

Подразумеваемые значения

Если в определения архетипов входят необязательные части, то полезно иметь возможность описывать «подразумеваемые» значения. Например, архетип понятия «измерение кровяного давления» может содержать необязательный фрагмент, описывающий положение пациента, который может принимать значение «лежа», «сидя» и «стоя». Поскольку эта часть класса ENTRY является необязательной, то данные могут быть созданы в соответствии с архетипом, не содержащим эту информацию. Однако при измерении кровяного давления пациент не может не находиться в определенном положении, поэтому с медицинской точки зрения разумно определить подразумеваемое или «допустимое» значение. Такое значение может быть явно описано в определении архетипа, чтобы все пользователи или системы знали, какое значение подразумевается в том случае, когда необязательные элементы не включены в состав данных. Подразумеваемые значения могут быть определены только на уровне листьев и могут быть заданы в классах *C_PRIMITIVE*.

Понятие подразумеваемых значений отличается от понятия «значений по умолчанию»; значения по умолчанию присутствуют в данных, а подразумеваемые — нет.

Пакет утверждений

Утверждения представляются в архетипах на языке логики предикатов первого порядка с контролем типов (FOL — first-order predicate logic). Они используются в двух случаях: 1) для описания ограничений слотов архетипа; 2) для описания инвариантов в ограничениях сложных объектов. В обоих случаях они ограничивают что-либо внутри архетипа. Ограничения внешних ресурсов, например терминологий, представляются в разделе связывания ограничений онтологии архетипа.

Конкретный синтаксис формулировок утверждений в архетипах совместим с языком ограничений объектов OCL организации OMG. Утверждения в архетипе представляют собой формулировки, содержащие следующие элементы:

- переменные, являющиеся именами атрибутов или путями ADL, заканчивающимися именами атрибутов (то есть они эквивалентны свойству класса ссылок в языке программирования);
- именованные константы любого примитивного типа данных плюс типы данных даты и времени;
- арифметические операторы: +, *, -, /, ^ (показатель степени);
- операторы отношения: >, <, >=, <=, =, !=, совпадения;
- булевские операторы: not, and, or, xor;
- кванторы, применяемые к переменным контейнера: for_all (для всех), exists (существует).

Пакет примитивов

В конечном счете любое определение архетипа развертывается до конечных узлов ограничений экземпляров примитивных типов данных. Пакет примитивов определяет семантику ограничений таких ти-

пов. Для большинства типов данных существуют, по крайней мере, два альтернативных способа описания ограничения; например, ограничение типа данных *C_DATE* может быть представлено в форме шаблона или интервала дат *Interval<Date>*.

Пакет онтологии

Все лингвистические и терминологические сущности архетипа представлены в разделе онтологии, семантика которого описана в пакете онтологии.

Онтология архетипа содержит следующие составляющие:

- список терминов, определенных локально в архетипе. Они идентифицируются кодами «atNNNN» и выполняют функцию идентификаторов узлов архетипа, с помощью которых создаются пути. В архетипе существует один такой список для каждого используемого естественного языка;

- список определений внешних ограничений, идентифицируемых кодами «acNNNN». В него включаются ограничения, определенные как внешние по отношению к архетипу, ссылки на которые даются с использованием экземпляра класса *CONSTANT_REF*. В архетипе существует один такой список для каждого используемого естественного языка;

- набор из одной или нескольких связей определений терминов с кодами терминов из внешних терминологий — факультативно;

- набор из одной или нескольких связей определений внешних ограничений с внешними источниками, например, с терминологиями — факультативно.

Глубина специализации

Любой конкретный архетип находится в определенной точке иерархии специализации архетипов. Глубина специализации указывается атрибутом *specialisation_depth*. Архетип, не являющийся специализацией другого архетипа, имеет глубину специализации 0. Коды терминов и ограничений, включенные в онтологию специализированных архетипов (т. е. тех, которых нет в онтологии родительского архетипа), определяются строго с использованием символа десятичной точки «.». Например, архетип, имеющий глубину специализации 2, будет использовать коды определений терминов следующего вида:

«at0.0.1» — новый термин, который определен в данном архетипе и не является специализацией какого-либо предшествующего термина в каком-либо родительском архетипе;

«at0001.0.1» — термин, являющийся специализацией термина «at0001», определенного в родителе верхнего уровня. Промежуточный «.0» необходим, чтобы показать, что новый термин имеет глубину 2, а не 1;

«at0001.1.1» — термин, являющийся специализацией термина «at0001.1», определенного в родителе промежуточного уровня, который в свою очередь специализирует термин «at0001», определенный в родителе верхнего уровня.

Данное систематическое определение кодов позволяет программному обеспечению использовать структуру кодов, чтобы делать логические выводы более быстро и точно относительно определений терминов вверх и вниз по иерархиям специализации. С другой стороны, коды ограничений не подчиняются этим правилам, а вместо этого принадлежат линейному пространству кодов.

Определения терминов и ограничений

Локальные определения терминов и ограничений моделируются в виде экземпляров класса *ARCHETYPE_TERM*, являющихся кодами, связанными со списком пар «имя-значение». Для любого определения термина или ограничения данный список должен содержать, по крайней мере, пары «имя-значение» для имен «text» (текст) и «description» (описание). Кроме того, список должен содержать такой элемент, как «provenance» (происхождение), который мог бы использоваться для указания того, что термин происходит из внешней терминологии. Атрибут *term_attribute_names* класса *ARCHETYPE_ONTOLOGY* содержит список имен атрибутов, используемых в архетипе для определений терминов и ограничений, включая «text» и «description», а также любые другие, используемые в разных местах.

Пакет обобщенных типов данных

Данный пакет включен для подтверждения семантики обобщенных типов данных, используемых в настоящем стандарте. Хотя типы List<T>, Set<T>, Hash<T,K> и Interval<T> являются обобщенными типами данных, поддерживаемыми многими программными средами, они не поддерживаются в языке UML непосредственно. В данном пакете новые типы данных, например List<String> (список строк), определяются с использованием зависимостей связей между новым базовым типом (в данном случае List<String>) и классом (LIST в данном примере), который определяет минимальную необходимую семантику для всех типов данных List.

Расширение, специфическое для предметной области (справочно)

Классы, специфические для предметной области, могут быть добавлены в модель ограничений архетипа с помощью наследования от класса *C_DOMAIN_TYPE*. В пункте 7.12.2 (научные/клинические вычислительные типы данных) показан общий подход, используемый для добавления классов ограничений общеупотребительных понятий, используемых в научных и клинических вычислениях, например «порядковое числительное», «кодированный термин» и «количество». В нем показаны типы ограничений *C_ORDINAL*, *C_CODED_TEXT* и *C_QUANTITY*, которые факультативно могут применяться в архетипах вместо используемой по умолчанию семантики ограничений, представленной с помощью экземпляров классов *C_OBJECT* / *C_ATTRIBUTE*.

Обзор языка ADL

Язык определения архетипов (ADL) является формальным языком описания архетипов. В языке ADL используются два разных синтаксиса — cADL (форма ограничений ADL) и dADL (форма определения данных ADL) для описания ограничений данных, являющихся экземплярами информационной модели, определенной в разделе 7.

Архетипы, описанные на языке ADL, имеют определенный синтаксис и напоминают файлы в языках программирования. Сам язык ADL имеет очень простой «связующий» синтаксис, в котором используются два других синтаксиса соответственно для представления структурированных ограничений и данных. Синтаксис cADL используется для описания определения архетипа, а синтаксис dADL — для описания данных, появляющихся в разделах «language (язык)», «description (описание)», «ontology (онтология)» и «revision_history (история версий)» ADL-архетипа. Структура верхнего уровня ADL-архетипа показана на рисунке 1 (сокращение FOPL обозначает логику предикатов первого порядка).

В разделе 8 определены синтаксисы dADL и cADL, синтаксис путей ADL, а также комбинированный синтаксис ADL, архетипы и библиотеки типов данных, специфичных для предметной области.

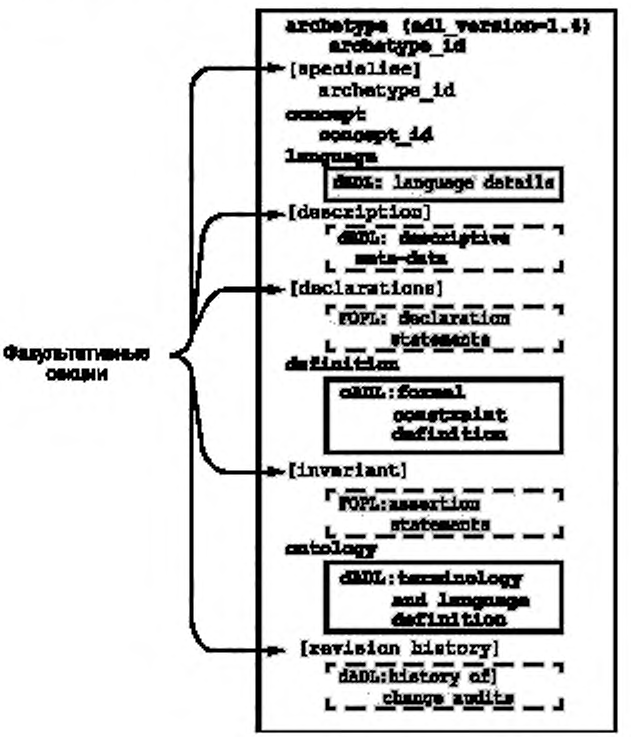

Рисунок 1 — Структура ADL-архетипа

Пример

Ниже приведен пример простого архетипа. Понятие *guitar* (гитара) определено в терминах ограничений общей модели понятия INSTRUMENT (инструмент). Имена, указанные в разделе «definition (определение)» (INSTRUMENT, size и т.д.), являются попеременно именами классов и атрибутов из модели объектов. Каждый блок квадратных скобок содержит спецификацию некоторого конкретного набора экземпляров, соответствующих конкретному понятию, например гитара или гриф. Эта спецификация определяется в терминах ограничений типов данных из обобщенной модели классов. Конечные пары квадратных скобок содержат ограничения примитивных типов данных, например, Integer, String и Boolean.

```
archetype (adl_version=1.4)
        adl-test-instrument.guitar.draft
concept
        [at0000]                                                    - - гитара
language
        original_language = <"ru">
        translations = <"de", ...>
definition
        INSTRUMENT[at0000] matches {
            size matches {|60..120| }                               - - размер в cm
            date_of_manufacture matches {yyyy-mm-??}
                        - - года и месяца достаточно
            parts cardinality matches {0..*} matches {
                PART[at0001] matches {                              - - гриф
                    material matches {[local::at0003]}              - - дерево
                }
            PART[at0002] matches {                                  - - корпус
                material matches {[local::at0003]}                  - - дерево

            }
        }
    }
ontology
    term_definitions = <
        [en] = <
            items = <
                ["at0000"] = <
                    text = <"гитара">;
                    description = <"струнный инструмент">
                >
                ["at0001"] = <
                    text = <"гриф">;
                    description = <"гриф гитары">
                >
                ["at0002"] = <
                    text = <"дерево">;
                    description = <"выдержанное мачтовое дерево">
                    >
                ["at0003"] = <
                    text = <"никелевый сплав">;
                    description = <"лады">
                >
            >
    >
    >
```

Медицинские примеры архетипов

Примечание — В разделе 8 приведено много примеров фрагментов кода на языке ADL, которые используются для иллюстрации характерных особенностей данного формализма. Они приведены только для иллюстративных целей и не должны рассматриваться как нормативные спецификации медицинских данных.

Не представляется возможным включить в настоящий стандарт полные медицинские примеры архетипов из-за их большого объема в печатном виде, но при желании ознакомиться с подборкой архетипов можно обратиться к следующему Интернет-сайту:

gost.e-ecolog.ru

На этом сайте опубликовано большое число архетипов, представленных как в формате ADL, так и в формате html. Представленные примеры включают переводы на разные языки и связи терминологий.

Примечание — Приведенная ссылка на Интернет-сайт работает только при указании префикса http.

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Информатизация здоровья

ПЕРЕДАЧА ЭЛЕКТРОННЫХ МЕДИЦИНСКИХ КАРТ

Часть 2

Спецификация передачи архетипов

Health informatics. Electronic health record communication. Part 2. Archetype interchange specification

Дата введения — 2013—07—01

1 Область применения

Настоящий стандарт определяет информационную архитектуру, необходимую для реализации интероперабельных обменов данными между системами и сервисами, которые используют или представляют данные ЭМК. Настоящий стандарт не предназначен для определения внутренней архитектуры или структуры базы данных подобных систем.

Субъектом медицинской карты или выписки из нее, подлежащей передаче, является отдельное лицо, а областью применения такой передачи преимущественно является оказание медицинской помощи данному лицу.

Использование медицинских карт для других целей, например административных, управленческих, для научных и эпидемиологических исследований, когда требуется агрегировать данные медицинских карт отдельных лиц, не входит в область применения настоящего стандарта, но для подобных вторичных приложений он может оказаться полезным.

Настоящий стандарт определяет модель архетипов, используемую для описания архетипов при передаче информации между фондами архетипов и между службами архетипов. Он определяет необязательное сериализованное представление, которое может быть использовано в качестве формата передачи отдельных архетипов. Подобная передача может быть организована, например, между библиотеками архетипов или между службой архетипов и службой, обеспечивающей постоянное хранение ЭМК или форматно-логический контроль ее данных.

2 Соответствие

Передача архетипа, используемого для ограничения части выписки EHR_EXTRACT, должна соответствовать информационной модели, определенной в разделе 7, а также факультативно может соответствовать спецификации языка определения архетипа ADL, определенной в разделе 8.

Настоящий стандарт не предписывает какое-либо конкретное представление архетипов для хранения в фонде архетипов, на сервере или в системе ведения ЭМК. Однако рекомендуется, чтобы используемое представление удовлетворяло требованиям, перечисленным в разделе 6.

3 Нормативные ссылки

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

ИСО 639 (все части) Коды для представления наименования языков [ISO 639 (all parts), Codes for the representation of names of languages]

Издание официальное

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

ИСО/МЭК 10646 Информационная технология. Универсальный многооктетный набор кодированных символов (UCS) [ISO/IEC 10646, Information technology — Universal Multiple-Octet Coded Character Set (UCS)]

4 Термины и определения

В настоящем стандарте применены следующие термины с соответствующими определениями:

4.1 **абстрактный класс** (abstract class): (в языке UML) «Виртуальный» общий родитель двух или более классов.

Примечание — Абстрактный класс не имеет экземпляров. В терминах моделирования он служит контейнером атрибутов и ассоциаций, которые могут принадлежать нескольким другим классам (его подклассам).

4.2 **экземпляр архетипа** (archetype instance): Отдельный экземпляр класса метаданных модели архетипов, определяющий клиническое понятие и ограничения значений, применяемые к одному классу экземпляров компонентов данных медицинской карты в выписке из ЭМК.

4.3 **модель архетипов** (archetype model): Информационная модель метаданных, описывающая характеристики записей электронной медицинской карты, специфичные для предметной области, путем определения значений или ограничений значений классов и атрибутов в базовой модели электронной медицинской карты.

4.4 **фонд архетипов** (archetype repository): Постоянное хранилище определений архетипов, доступ к которому осуществляется с помощью клиентского средства разработки архетипов или компонента времени исполнения службы ведения электронных медицинских карт.

4.5 **регистрационный журнал** (audit trail): Хронологическая регистрация действий пользователей информационной системы, обеспечивающая возможность достоверного восстановления предыдущих состояний информации.

4.6 **понятие** (concept): Единица знаний, созданная с помощью уникальной комбинации характеристик (определение 3.2.1 из [6]).

Примечание — Понятия не обязательно связаны с конкретными языками. Однако на них влияет социальная или культурная среда, часто приводящая к разным категоризациям.

4.7 **электронная медицинская карта** (electronic health record): Хранилище информации, относящейся к здоровью субъекта медицинской помощи, в форме, пригодной для компьютерной обработки.

Примечание — Адаптировано из [10], определение 2.11.

4.8 **запись электронной медицинской карты** (electronic health record entry): Любые данные из медицинской карты.

Примечание — Результаты клинических исследований, заключения, рассуждения, намерения, планы или действия без конкретного определения их формального представления, иерархической организации или конкретного класса (классов) элементов медицинской карты, которые могут быть использованы для их представления.

4.9 **выписка из электронной медицинской карты** (electronic health record extract): Вся электронная медицинская карта субъекта медицинской помощи или ее часть, передаваемая в соответствии с комплексом стандартов ИСО 13606.

4.10 **система ведения электронных медицинских карт** (electronic health record system): Система, предназначенная для записи, извлечения и обработки информации в электронных медицинских картах.

4.11 **типовой** (generic): Применимый к требованиям или информационным моделям разных медицинских профессий, предметных областей и стран.

4.12 **метаданные** (metadata): Данные, определяющие и описывающие другие данные (определение 3.2.18 из [7]).

4.13 **пациент** (patient): Субъект медицинской помощи.

4.14 **семантический** (semantic): Относящийся к смыслу языковой конструкции.

4.15 **семантическая интероперабельность** (semantic interoperability): Свойство данных, совместно используемых несколькими системами, быть понятными на уровне полностью определенных понятий предметной области (определение 3.38 из [9]).

4.16 **интегрированная электронная медицинская карта** (shareable electronic health record): Электронная медицинская карта, соответствующая стандартизованной информационной модели, не зависящая от систем ведения электронных медицинских карт и доступная многим авторизованным пользователям.

4.17 **субъект медицинской помощи** (subject of care): Лицо, которому запланирована медицинская помощь, получающее или получившее медицинскую помощь (адаптировано из [5]).

5 Сокращения

В настоящем стандарте использованы следующие сокращения:

ADL — язык определения архетипов (Archetype Definition Language);

ЭМК — электронная медицинская карта (Electronic Health Record, EHR);

OPO — открытая распределенная обработка (Open Distributed Processing, ODP);

OWL — веб-язык онтологий (Ontology Web Language);

UML — унифицированный язык моделирования (Unified Modelling Language);

XML — расширяемый язык разметки (Extensible Mark-up Language).

6 Требования к представлению архетипов

6.1 Общие сведения

В данном разделе приведен список формальных требований к представлению архетипов. Они служили основой для разработки модели архетипов, определенной в разделе 7. Эти требования необходимо было включить в настоящий стандарт, поскольку существует лишь небольшое число опубликованных работ, посвященных требованиям к такой модели, в отличие от самой ЭМК, для которой был принят стандарт ИСО/ТС 18308.

6.2 Определение, описание архетипа и информация о его публикации

6.2.1 Определение архетипа должно содержать перечисленную ниже информацию.

6.2.1.1 Глобально уникальный идентификатор определения данного архетипа.

6.2.1.2 Идентификатор фонда, в котором данный архетип был создан или теперь главным образом содержится, или уполномоченной организации, ответственной за сопровождение данного архетипа. Этот фонд должен быть тем, который отвечает за состояние окончательной публикации данного архетипа.

6.2.1.3 Понятие, которое наилучшим образом определяет общую клиническую область применения экземпляров классов, соответствующих данному архетипу в целом, выраженное в виде кодированного термина или свободного текста на заданном естественном языке.

6.2.1.4 Область информатизации здоровья, в которой применяется данный архетип (например ЭМК). Данная область отображается на множество базовых моделей, с которыми может использоваться данный архетип.

6.2.1.5 Основная базовая модель, для которой данный архетип был идеально приспособлен.

Примечание — В данной области информатизации здоровья архетип может быть пригоден для использования с несколькими базовыми моделями, но предполагается, что архетип оптимизирован только для одной базовой модели.

6.2.1.6 Естественный язык, на котором данный архетип был первоначально определен, заданный его кодом по ИСО 639. В случае неточного перевода для интерпретации архетипа должно использоваться определение на этом языке.

6.2.2 В тех случаях, когда это возможно, определение архетипа может содержать перечисленную ниже информацию.

6.2.2.1 Глобально уникальный идентификатор архетипа, специализацией которого является данный архетип и которому он также должен соответствовать.

6.2.2.2 Глобально уникальный идентификатор прежнего архетипа, заменяемого данным определением, если только данная версия архетипа не первична.

6.2.2.3 Причина создания новой версии ранее существовавшего архетипа.

6.2.2.4 Идентификатор того архетипа, на который данный архетип был заменен (если таковая замена имела место).

Примечание — В фонде архетипов с контролем версий добавление данной информации может быть осуществлено только по ссылке; способ реализации данного механизма не входит в область применения настоящего стандарта.

6.2.2.5 Архетип должен иметь один или несколько наборов описаний, характеризующих его использование и назначение. В определение архетипа может быть включено много версий данной информации, представленной на разных естественных языках или предназначенной для разных типов потенциальных пользователей.

6.2.3 Набор описаний архетипа должен содержать перечисленную ниже информацию.

6.2.3.1 Однозначно идентифицированная сторона, ответственная за предоставление данного набора описаний. Ее идентификация может факультативно включать в себя название организации, которую представляет данная сторона, или название органа, от имени которого она действует. Дополнительно может быть указана контактная информация этой стороны.

6.2.3.2 Естественный язык, на котором представлен данный набор описаний, указанный его кодом в стандарте ИСО 639.

6.2.3.3 Формальное заявление, описывающее область применения и клиническое назначение данного архетипа, выраженное в виде кода термина или свободного текста на данном естественном языке.

Для облегчения поиска требуемых архетипов в фонде рекомендуется выражать эти характеристики в форме кодированного термина.

Пример — Клиническая область применения и назначение могут описывать:

- основную медицинскую специальность, для которой предназначен архетип;

- список клинических, медицинских или процедурных терминов (ключевых слов): диагнозы, действия, лекарства, исследования и т. д.;

- тип пациента, для которого предназначен архетип (возраст, пол и т. д.).

6.2.4 Набор описаний архетипа может при необходимости содержать представленную ниже информацию.

6.2.4.1 Формальное заявление о цели применения архетипа.

Примечание — В идеале таким формальным заявлением должно быть кодированное выражение, однако соответствующая терминология для него пока еще не разработана.

6.2.4.2 Формализация ситуаций, в которых пользователи могут ошибочно решить, что надо использовать данный архетип. В ней могут быть также оговорены те виды базовой модели, для которых данный архетип не подходит.

6.2.4.3 Подробное пояснение назначения данного архетипа, включая любые характеристики, представляющие особый интерес, или комментарии. Сюда может относиться указание групп лиц, которым это пояснение предназначено, например студентам. Данная информация может включаться непосредственно и/или по ссылке (например можно задать унифицированный указатель ресурсов URL).

6.2.4.4 Описание, связь или ссылка на опубликованные медицинские знания, на основании которых создано определение данного архетипа.

6.2.5 Определение архетипа должно содержать формулировку состояния его публикации.

Определение архетипа может проходить ряд состояний публикации, например процесс утверждения, оставаясь неизменным. Для целей аудита такие последовательные состояния должны сохраняться как часть архетипа. Однако изменение состояния публикации архетипа само по себе не должно служить поводом для формальной смены идентификатора, который используется для ссылки на данный архетип в выписке EHR_EXTRACT, поскольку спецификация ограничения при этом не изменяется.

6.2.6 Сведения о состоянии публикации архетипа должны включать в себя перечисленную ниже информацию.

6.2.6.1 Состояние публикации архетипа, выбранное из следующего списка:

- тестовый/демонстрационный;
- предварительный;
- черновой;
- для внутренних целей;

- общедоступный;
- предпочтительный;
- отмененный.

6.2.6.2 Дата присвоения данного состояния публикации.

Примечание — Датой первого состояния публикации архетипа будет дата первичного составления документа.

6.2.6.3 Уникальный идентификатор стороны, отправившей данный архетип в фонд и тем самым утвердившей данное состояние публикации. Это идентификатор может быть дополнен названием организации, которую представляет эта сторона.

6.2.6.4 Уникальный идентификатор органа, санкционировавшего данное изменение состояния публикации.

6.2.6.5 Ожидаемая дата проверки текущего статуса публикации и содержания самого архетипа в целях подтверждения, что они остаются действительными с клинической точки зрения.

6.3 Ограничения узлов архетипа

6.3.1 Общие сведения

Определение архетипа включает в себя спецификацию иерархической схемы, которой должны соответствовать экземпляры данных (например данных ЭМК). Эта схема определяет иерархическую организацию совокупности узлов, отношения между ними и ограничения допустимых значений атрибутов и данных. Кроме того, она должна соответствовать той базовой модели (моделям), для которой применимо данное определение.

6.3.2 Ссылки на узлы архетипа

6.3.2.1 Любой узел в иерархии узлов архетипа может быть определен в явном виде или по ссылке, либо может быть указан как часть уже существующего архетипа или весь такой архетип.

6.3.2.2 Ссылка на уже существующий архетип или его фрагмент может быть задана в явной форме в виде идентификатора архетипа, который может быть дополнен указанием узла, соответствующего точке вставки фрагмента архетипа.

6.3.2.3 Ссылка на фрагмент архетипа может быть внутренней (т.е. указывать на часть текущего архетипа).

6.3.2.4 Узел архетипа может быть задан как один из совокупности возможных архетипов посредством указания явного списка кандидатов и/или задания набора ограничений конкретных атрибутов определения архетипа.

6.3.2.5 В дополнение к указанию одного или нескольких фрагментов архетипа по ссылке или с помощью ограничения должна быть обеспечена возможность добавить пояснение причины, по которой эти фраменты указаны в данной точке иерархии узлов рассматриваемого архетипа.

6.3.3 Спецификация узла архетипа

6.3.3.1 Спецификация узла архетипа (если он определен не по ссылке) должна содержать перечисленную ниже информацию.

6.3.3.2 Внутренне уникальный идентификатор данного узла архетипа. В сочетании с глобально уникальным идентификатором определения данного архетипа он должен быть глобально уникальной ссылкой на этот узел.

6.3.3.3 Класс в иерархии экземпляров, определенной в той базовой модели, для которой данный архетип был идеально приспособлен. Экземпляр этого класса должен соответствовать данному узлу архетипа. Для иерархии данных ЭМК, соответствующей настоящему стандарту, данный класс должен быть выбран из числа следующих:

- FOLDER (том);
- COMPOSITION (композиция);
- SECTION (раздел);
- ENTRY (подраздел);
- CLUSTER (кластер);
- ELEMENT (элемент).

Кратность экземпляров этого класса, выраженная в виде диапазона, которые могут быть созданы в соответствии с определением данного узла архетипа в иерархии экземпляров.

6.3.3.4 Для управления созданием экземпляров, соответствующих данному узлу архетипа, могут быть заданы другие ограничения и правила.

6.3.3.5 Правила ограничений могут быть выражены в форме логических условий и содержать ссылку на параметры среды, например на текущее время, на место или на участников, либо могут быть связаны с уже существующими значениями других узлов в иерархии экземпляров. Правила ограничений могут использоваться для представления связей между данными ЭМК и рабочими процессами или клиническими маршрутами.

6.3.3.6 Правила ограничений могут быть выражены в форме критериев включения или исключения.

6.3.3.7 Любое определение правила ограничения должно идентифицировать формализм (включая версию), в котором оно выражено (например ADL, OWL).

6.3.4 Связь узлов архетипа с терминами

6.3.4.1 Каждый узел иерархической схемы архетипа должен быть ассоциирован по крайней мере с одним медицинским термином, наиболее точно выражающим понятие, которое должно быть представлено данным узлом при реализации соответствующей иерархии экземпляров. Обычно данный термин должен быть включен в экземпляр или на него должна быть ссылка.

6.3.4.2 В целях поиска архетипа в фонде или поиска соответствующих экземпляров любой узел архетипа может быть отображен на произвольное число дополнительных понятий, терминов и синонимов из терминологических систем.

6.3.4.3 При отображении любого понятия на термин или текст должно быть указано назначение этого отображения с использованием значения из следующего списка:

- основное понятие;
- связывание терминов;
- синоним;
- перевод на другой язык.

6.3.4.4 Любая ссылка на кодированный термин должна содержать код, рубрику и систему кодирования (включая ее версию), из которой взяты данные код и рубрика. Кроме того, должна существовать возможность указать естественный язык, на который данный термин был отображен или на который был сделан перевод.

6.3.5 Ограничения атрибутов и ассоциаций

В узле архетипа могут быть указаны ограничения любых атрибутов или ассоциаций, соответствующих атрибутам и ассоциациям данного узла, описанным в основной базовой модели.

Такие ограничения могут заранее задавать или ограничивать некоторую часть или всю контекстную информацию, содержащуюся в данном экземпляре, в соответствии с представлением в базовой модели. Контекстная информация, например лицо, с которым связано конкретное исследование или заключение, формально представлена в большинстве ЭМК-подобных моделей в целях обеспечения реализации безопасных запросов и выборок информации, даже если эта информация может быть определена по имени архетипа или фасету терминологической системы, из которой берутся значения данных. Некоторые архетипы или их фрагменты заранее определяют элементы контекстной информации, которые могут быть включены в определение архетипа (например, чтобы указать в архетипе семейного анамнеза, что субъектом информации является родственник, а не пациент). Применимость любого аспекта контекста к данному узлу архетипа определяется атрибутами контекста соответствующего узла базовой модели целевой ЭМК. Например, в базовой модели, определенной в EN 13606, субъект информации представлен на уровне элемента Entry. Между совокупостями элементов контекстной информации, определенных в разных базовых моделях, существует значительная общность (например между моделями, определенными в EN 13606 и HL7 CDA Release 2). В идеале должен быть принят общий набор меток этих элементов, что позволит применять соответствующие контекстные ограничения к нескольким базовым моделям.

6.3.6 Дополнительная информация

6.3.6.1 Необходима возможность указания перечисленной ниже информации для любого атрибута или любой ассоциации заданной базовой модели.

6.3.6.2 Имя атрибута или ассоциации, к которым применяется данное ограничение. Оно должно быть взято из той базовой модели, для которой данный архетип был идеально приспособлен.

6.3.6.3 Общая метка для данного аспекта контекста. Для архетипов ЭМК это значение должно быть выбрано из (предварительного) списка значений, приведенных в таблице 1.

6.3.6.4 Обязательность включения атрибута или ассоциации, соответствующих данному аспекту контекста, в правильный экземпляр ЭМК для заданной базовой модели.

6.3.6.5 Число экземпляров (выраженное в форме диапазона), соответствующих данному аспекту контекста, которые могут быть созданы.

6.3.6.6 Если допускается множество экземпляров, то должна существовать возможность указать, должны ли они быть представлены как упорядоченный или неупорядоченный список.

6.3.6.7 Если допускается множество экземпляров, то должна существовать возможность указать, должны ли быть уникальными соответствующие значения данных (конечных узлов или атрибутов).

6.3.6.8 Ограничения могут быть указаны для значений данных конечных узлов или конечных атрибутов.

6.3.6.9 Дополнительно могут быть заданы другие ограничения и правила для того, чтобы управлять созданием экземпляров, соответствующих атрибуту или ассоциации базовой модели.

6.4 Ограничения значений данных

6.4.1 Должна существовать возможность задавать ограничения и правила для значений данных конечных узлов в иерархии базовой модели, или для любых других атрибутов любого узла архетипа.

6.4.2 Должна существовать возможность задавать перечисленную ниже информацию об ограничениях значений данных.

6.4.2.1 Разрешено ли данным принимать пустое значение, и дополнительно указать, почему оно пустое (например указать код причины пустого значения).

6.4.2.2 Основано ли ограничение или правило на критерии включения или исключения.

6.4.2.3 Формализм (включая версию), в котором представлено данное ограничение.

6.4.2.4 Предлагаемое фиксированное (предписанное) значение экземпляров, соответствующих определению архетипа.

6.4.2.5 Предлагаемое значение по умолчанию для экземпляров, соответствующих определению архетипа.

6.4.2.6 Список допустимых значений экземпляров, соответствующих определению архетипа. Этот список должен быть подмножеством допустимых значений, определенных в используемой базовой модели.

6.4.3 Для количественных типов данных должна существовать возможность указать следующие параметры:

- диапазон допустимых значений экземпляров, соответствующих определению архетипа;
- диапазон, в котором значения считаются выходящими за пределы нормы или критичными с клинической точки зрения;
- предлагаемые единицы измерения значений экземпляров, соответствующих определению архетипа.

6.4.4 Для типов данных даты и даты и времени должна существовать возможность указать следующие параметры:

- диапазон допустимых значений экземпляров, соответствующих определению архетипа;
- предлагаемые единицы измерения значений экземпляров, соответствующих определению архетипа.

6.4.5 Для текстовых типов данных должна существовать возможность указать следующие параметры:

- шаблон строки, задающий диапазон возможных значений;
- предлагаемая система кодирования, из которой берутся значения экземпляров, соответствующих определению архетипа.

6.4.6 Правила ограничений могут быть выражены в форме логических условий и могут включать ссылку на параметры среды, например на текущее время, на место или на участников, либо могут быть связаны с уже существующими значениями других узлов в иерархии экземпляров.

6.4.7 Ссылка на существующее значение должна определять данный экземпляр точно и однозначно. Например, может потребоваться включить ссылку на следующие элементы:

- идентификатор архетипа;
- идентификатор узла архетипа;
- имя атрибута или ассоциации;
- вхождение в иерархию архетипов, например: первое, самое последнее, любое, *n*-ое упорядоченное по параметру *y* (*n*-й элемент множества экземпляров, упорядоченных по параметру *y*), наибольшее значение, наименьшее значение, один или несколько экземпляров в (определяемом) недавнем интервале времени;

- предлагаемую связь между данным значением указанного экземпляра и ограничиваемым значением, например: то же самое значение; подмножество или подстрока; больше чем, больше или равно, меньше чем, меньше или равно; раньше чем, позже чем и т. д.; если ..., то ...; не должно быть тем же самым.

6.4.8 Для формирования составных правил эти сравнительные ограничения могут быть вложенными и включать логические операторы или операторы над множествами.

6.5 Профиль по отношению к базовой модели EN 13606-1

В приведенных ниже таблицах определен типовой набор меток контекста, используемых в базовой модели ЭМК EN13606-1 для указания связи наблюдаемых данных, ожидаемых данных и заключений с субъектом медицинской помощи или с другими частями ЭМК этого субъекта. В таблице 1 приведены разделы контекста, а в таблице 2 — отображение этих разделов на базовую модель EN 13606.

Таблица 1 — Список разделов и элементов контекста

Раздел контекста	Элемент контекста	Описание
Meaning (Смысл)	Meaning (Смысл)	Формальное понятие, определяющее объект ЭМК, который должен быть реализован в данном узле ЭМК
Subject of information (Субъект информации)	Subject of information (Субъект информации)	Правила для допустимых значений субъекта информации, например указание того, что субъект информации не может быть пациентом или должен быть кровным родственником
Act status (Статус действия)	Act status (Статус действия)	Правила для допустимых значений статуса действия, например указание того, что действием должна быть запланированная/предписанная деятельность
Temporal relationship (Привязка ко времени)	Temporal relationship (Привязка ко времени)	Определение связи информации со временем ее регистрации, например предшествующая, текущая, будущая
Structure (Структура)	Structure (Структура)	Пространственная структура, используемая для представления (предоставления) структуры данных, например список, таблица, дерево
Observation time (Время исследования)	Observation time (Время исследования)	Правила для допустимых значений времени (или временного интервала), когда данный комплекс исследований предлагалось провести или когда он был действительно проведен
Link (Связь)		Большинство связей (LINK в стандарте EN 13606, Act Relationship в стандарте HL7) определяются по факту в отдельных экземплярах данных ЭМК. Однако могут быть случаи, когда конкретные виды клинических данных должны иметь определенные связи заранее. Например, для некоторых лечебных воздействий всегда может требоваться ссылка на документ информированного согласия, уже существующий в ЭМК, или на уже существующие данные клинического исследования, обосновывающие это воздействие
	Nature (Природа)	Вид связи (nature в стандарте CEN, код класса Act Relationship в стандарте HL7), который должен или может быть скомпонован
	Role (Роль)	Роль целевого объекта связи
	Follow_link (Следовать связи)	Правила для случая, когда требуется, чтобы связь прослеживалась при извлечении исходного или целевого объекта связи из системы ведения ЭМК (follow_link в стандарте CEN, separableInd в стандарте HL7)

Окончание таблицы 1

Раздел контекста	Элемент контекста	Описание
Participation (Участие)		Если некоторые участники могут или должны быть определены в экземпляре узла ЭМК
	Function (Функция)	Функциональная роль участника
	Mandatory attestation (Обязательная аттестация)	Указывает, должен ли данный участник аттестовать экземпляр; это требование не содержит указания, аттестуется ли данный узел отдельно, или он может быть аттестован как часть большей совокупности узлов ЭМК, например на уровне документа
	Attestation reason (Причина аттестации)	Определяет фиксированную причину аттестации, например аттестация осуществляется для выполнения конкретной юридической функции

Таблица 2 — Базовая модель ЕН 13606 — Профиль контекста

Класс базовой модели	Раздел контекста	Соответствующий атрибут базовой модели
FOLDER	Meaning	Meaning
	Link	LINK
COMPOSITION	Meaning	Meaning
	Link	LINK
	Participation	composer other_participations
SECTION	Meaning	Meaning
	Link	LINK
ENTRY	Meaning	Meaning
	Link	LINK
	Participation	other_participations
	Subject of information	subject_of_information
	Act status	act_status
CLUSTER	Meaning	Meaning
	Link	LINK
	Structure	structure_type
	Observation time	obs_time
ELEMENT	Meaning	Meaning
	Link	LINK
	Observation time	obs_time

7 Модель архетипов

7.1 Введение

7.1.1 Общие сведения

Данная модель представлена с использованием ограниченной формы диаграмм UML, описанной ниже в профиле UML.

7.1.2 Профиль UML

Классы модели вместе со своими ассоциациями и наследованием группируются в пакеты, представленные ниже на отдельных диаграммах. Пакеты ограничены прямоугольниками. Над левым верхним углом прямоугольника, ограничивающего пакет, указано наименование пакета, также заключенное в прямоугольник.

Прямоугольники классов обычно содержат три части.

Первая часть содержит имя класса, представленное прописными буквами. Кроме того, она может содержать в круглых скобках имя класса-владельца, если данный класс принадлежит пакету, который не представляет данная диаграмма. На некоторых диаграммах в верхней части также указаны ограничения данного класса.

Вторая часть, если она присутствует, содержит атрибуты с указанием имени, типа и кратности атрибута. Кратность может также быть уточнена маркером «ordered (упорядоченный)». Имена атрибутов написаны строчными буквами. Имена типов данных атрибутов записаны с первой прописной буквы, остальные строчные, если данный тип является одним из базовых типов, или все прописные, если данный тип представляет собой другой класс.

Третья часть, если она присутствует, содержит операции с указанием имени операции, типа возвращаемых данных и передаваемых параметров. Написание имен операций и типов данных подчиняется тем же правилам, что и для атрибутов.

Прямоугольник класса только с двумя частями содержит имя класса и атрибуты, а прямоугольник с одной частью содержит только имя класса.

Для повышения наглядности линии наследования показаны сплошными линиями, а линии связей — пунктирными. Затенение прямоугольников классов иногда используется для выделения конкретных группировок классов. Наименование пакета, написанное не полужирным, а обычным шрифтом означает, что данный пакет представлен на отдельной диаграмме.

Ассоциации классов всегда однонаправленные, при этом наименование ассоциации и кратность помещены на конце линии ассоциации. Если требуется двунаправленная ассоциация, то она показывается в виде пары однонаправленных ассоциаций двух классов, направленных в разные стороны. Данное ограничение призвано обеспечить возможность автоматического документирования каждой ассоциации, как если бы она была атрибутом класса, расположенного в конце линии ассоциации. Стрелки навигации не используются.

7.1.3 Подробное документирование модели

Порядок документирования — по пакетам, а внутри пакета — по классам.

Описание каждого класса имеет начальный раздел, в котором указан владеющий пакет, каждое наследование, внутренние элементы и документирование внутренней модели. За ним следуют четыре раздела в табличной форме, описывающие:

a) атрибуты;

b) атрибуты, унаследованные от ассоциаций;

c) операции;

d) ограничения.

Сами ассоциации показаны на диаграммах с использованием нотации UML, но документируются как унаследованные атрибуты с использованием следующих соглашений:

Имя на дальнем конце ассоциации	Преобразуется в	Имя атрибута
Класс на дальнем конце ассоциации	Преобразуется в	Тип данных атрибута

Кратность ассоциации	Генерирует	Тип контейнера	И	Обязательность атрибута	Исходная кратность
0..*		Set<КЛАСС на дальнем конце >		0..1	0..*
0..* {ordered}		List<КЛАСС на дальнем конце>		0..1	1..* {ordered}
1..*		Set<КЛАСС на дальнем конце>		1	1..*
1..* {ordered}		List<КЛАСС на дальнем конце>		1	1..* {ordered}
*		Set<КЛАСС на дальнем конце>		0..1	*
0..1		Не контейнер		0..1	Не применима
1		Не контейнер		1	Не применима

7.1.4 Структура пакетов

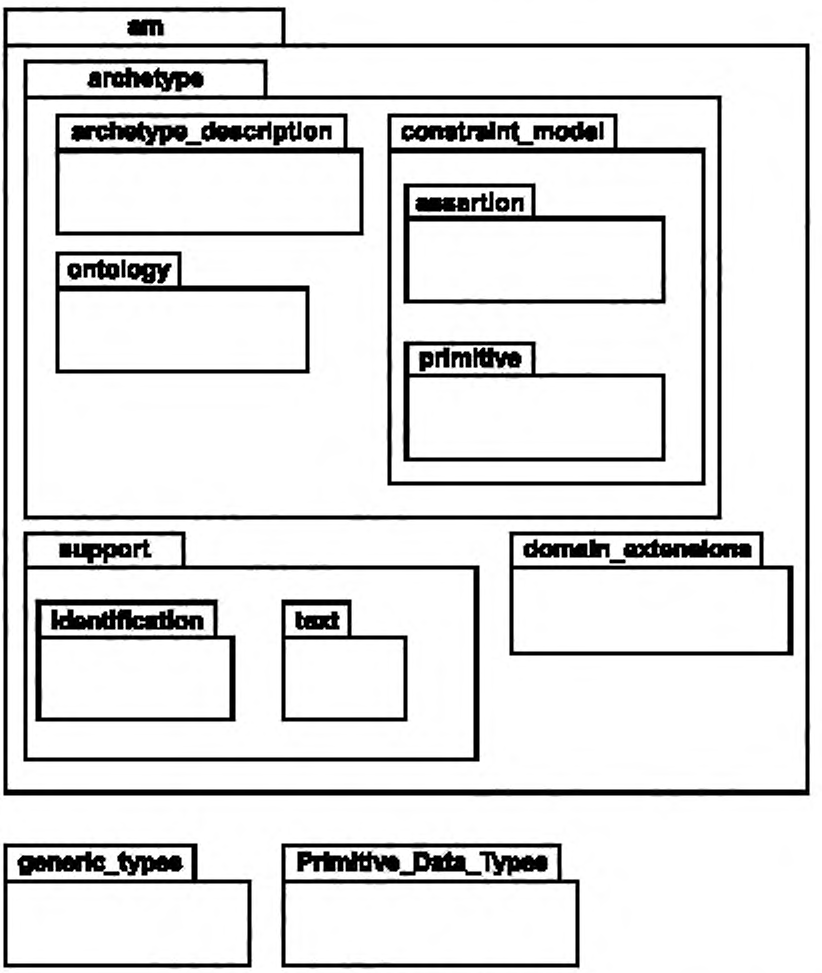

Рисунок 2 — Структура пакетов

Общая модель архетипа, показанная на рисунках 3 и 4, определяет обобщенное представление архетипов для целей интероперабельности и обмена данными.

7.2 Обзор

7.2.1 Общая информация

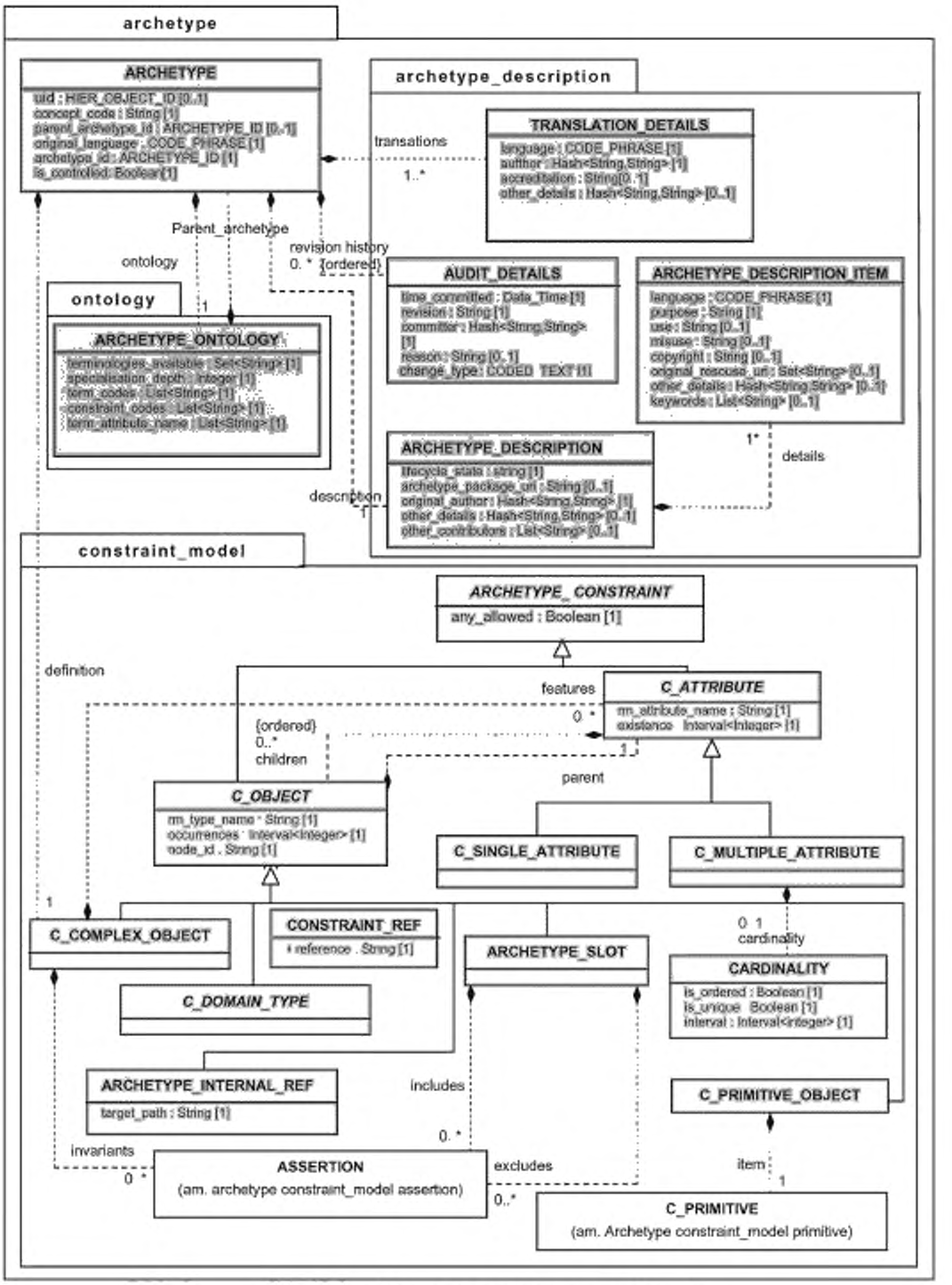

Рисунок 3 — Общий вид модели архетипа. Часть 1

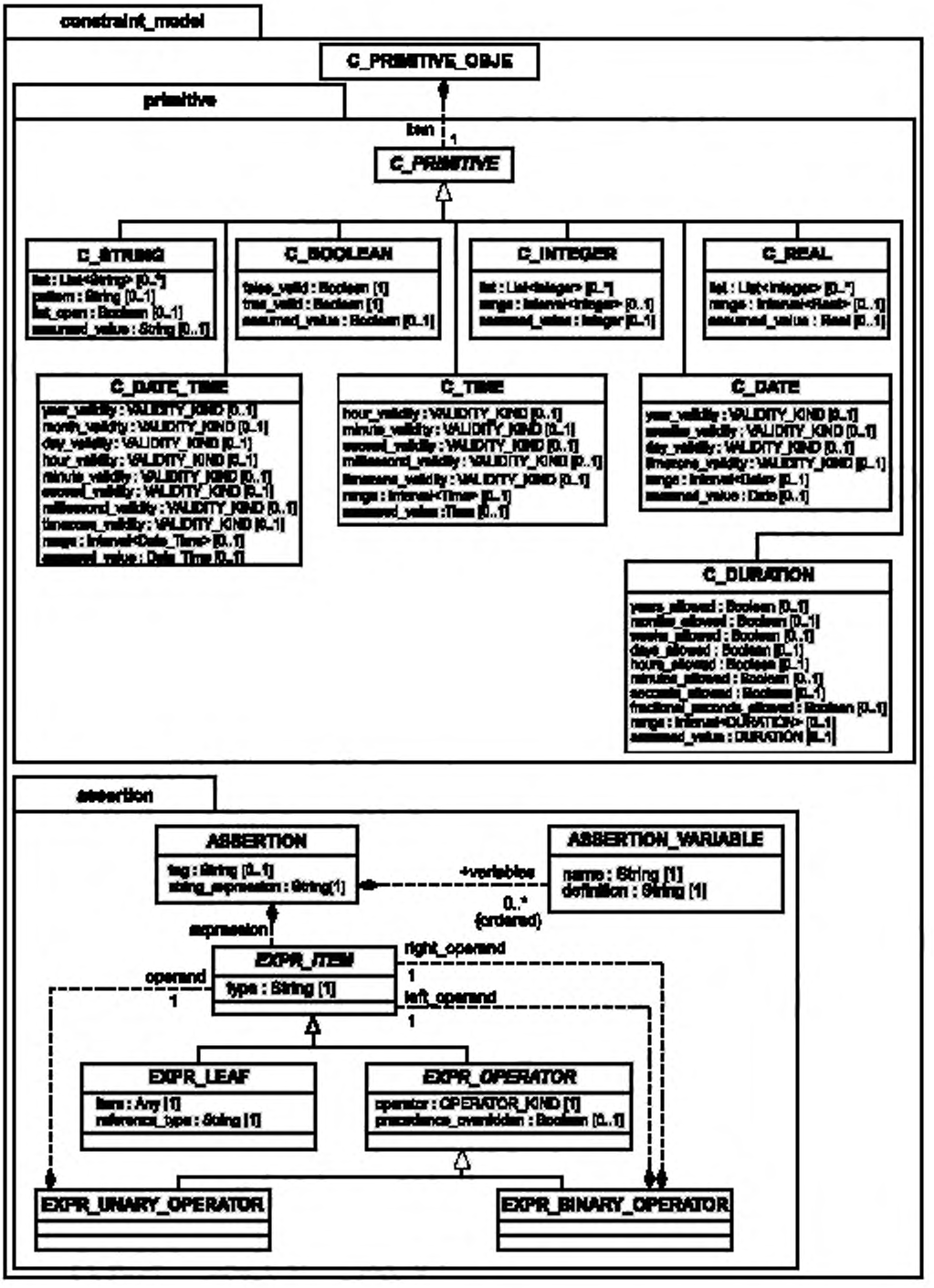

Рисунок 4 — Общий вид модели архетипа. Часть 2

Документация, сгенерированная из модели UML

Примечание — Приведенный ниже список имен пакетов и классов включен для того, чтобы показать высокоуровневый общий вид модели в том виде, как она представлена в данном подразделе.

- am
  - archetype
    - ARCHETYPE
    - archetype_description
      - ARCHETYPE_DESCRIPTION
      - ARCHETYPE_DESCRIPTION_ITEM
      - AUDIT_DETAILS
      - TRANSLATION_DETAILS
    - constraint_model
      - ARCHETYPE_CONSTRAINT
      - ARCHETYPE_INTERNAL_REF
      - ARCHETYPE_SLOT
      - C_ATTRIBUTE
      - C_COMPLEX_OBJECT
      - C_DOMAIN_TYPE
      - C_MULTIPLE_ATTRIBUTE
      - C_OBJECT
      - C_PRIMITIVE_OBJECT
      - C_SINGLE_ATTRIBUTE
      - CARDINALITY
      - CONSTRAINT_REF
    - assertion
      - ASSERTION
      - ASSERTION_VARIABLE
      - EXPR_BINARY_OPERATOR
      - EXPR_ITEM
      - EXPR_LEAF
      - EXPR_OPERATOR
      - EXPR_UNARY_OPERATOR
    - primitive
      - C_BOOLEAN
      - C_DATE
      - C_DATE_TIME
      - C_DURATION
        - C_INTEGER
      - C_PRIMITIVE
      - C_REAL
      - C_STRING
      - C_TIME
    - ontology
      - ARCHETYPE_ONTOLOGY
      - ARCHETYPE_TERM
    - domain_extensions
      - C_CODED_TEXT
      - C_ORDINAL
      - C_QUANTITY
      - C_QUANTITY_ITEM
      - ORDINAL
    - support
      - OPERATOR_KIND
      - VALIDITY_KIND
      - Identification
        - ARCHETYPE_ID
        - HIER_OBJECT_ID
        - OBJECT_ID
        - TERMINOLOGY_ID

text
CODE_PHRASE
CODED_TEXT
TERM_MAPPING
TEXT

7.2.2 Пакет :: am

Внутренние элементы

Имя	Тип
archetype	Пакет (package)
domain_extensions	Пакет (package)
support	Пакет (package)

7.3 Пакет archetype

7.3.1 Общая информация

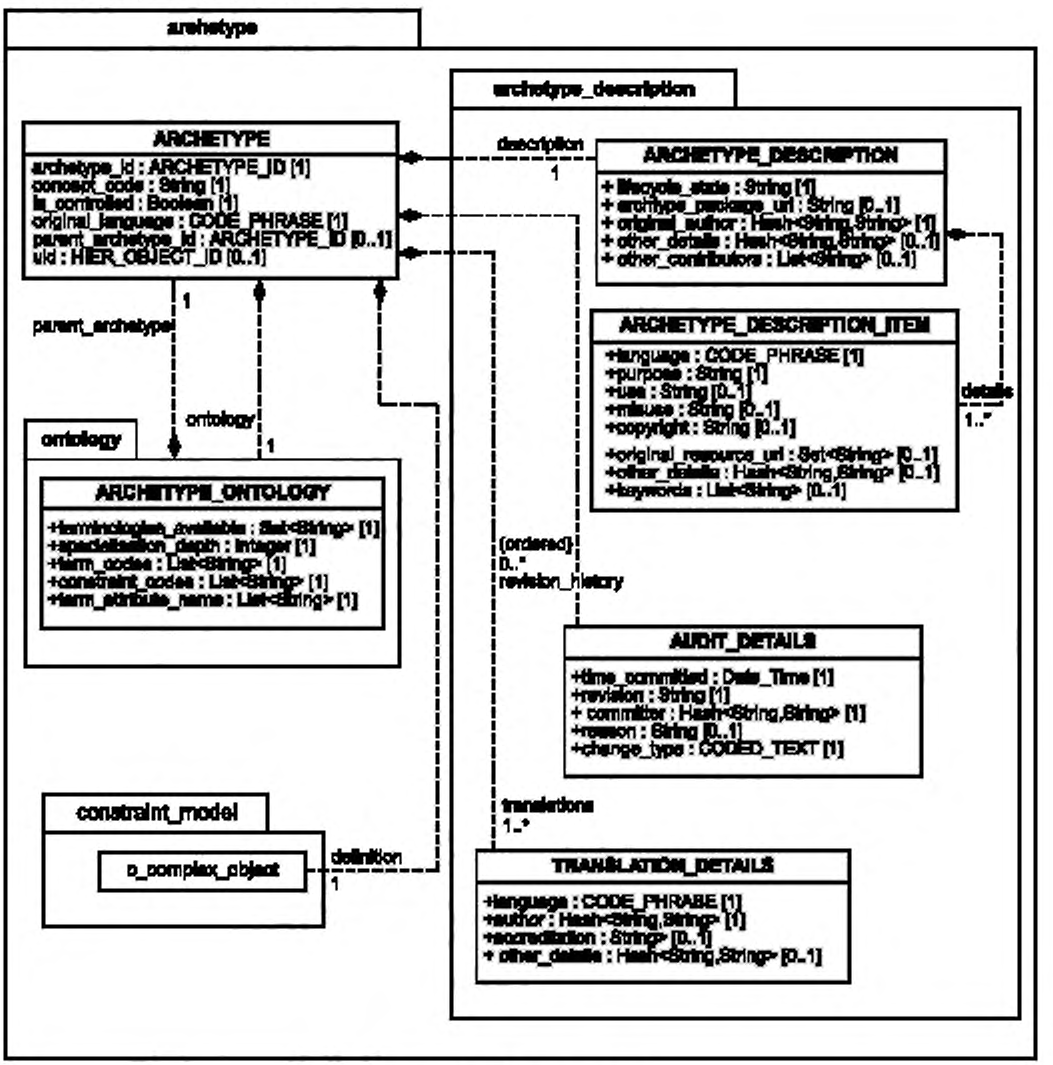

Рисунок 5 — Пакет archetype

7.3.2 Пакет :: archetype
Внутренние элементы

Имя	Тип
ARCHETYPE	Класс (class)
archetype_description	Пакет (package)
constraint_model	Пакет (package)
ontology	Пакет (package)

Пакет: archetype
Класс ARCHETYPE
Основной класс пакета archetype.
Атрибуты

Сигнатура	Обязательность	Кратность	Описание
archetype_id : ARCHETYPE_ID	1	--	Фасетный идентификатор данного архетипа в пространстве архетипов
concept_code : String	1	--	Нормативное назначение данного архетипа в целом
is_controlled : Boolean	1	--	Истина, если данный архетип подлежит контролю изменений; в этом случае создается история версий
original_language : CODE_PHRASE	1	--	Язык, на котором данный архетип был впервые описан
parent_archetype_id : ARCHETYPE_ID	0..1	--	Идентификатор специализируемого родителя даного архетипа
uid : HIER_OBJECT_ID	0..1	--	Объектный идентификатор (OID) данного архетипа

Атрибуты, унаследованные от ассоциаций

Сигнатура	Обязательность	Кратность	Описание
revision_history : List<AUDIT_DETAILS>	0..1	0..* ordered	История версий архетипа; требуется, только если атрибут is_controlled = True
description : ARCHETYPE_DESCRIPTION	1	--	Описание архетипа и информация о его жизненном цикле
ontology : ARCHETYPE_ONTOLOGY	1	--	Онтология архетипа
definition : C_COMPLEX_OBJECT	1	--	Корневой узел данного архетипа
translations : Set<TRANSLATION_DETAILS>	1	1..*	Перечень деталей каждого перевода на естественный язык, включенного в данный архетип

Ограничения

Имя	Выражение
revision_history_validity	inv: is_controlled implies (revision_history <> Void and revision_history.is_empty)
archetype_id_validity	inv: archetype_id <> Void
description_exists	inv: description <> Void
ontology_exists	inv: ontology <> Void
definition_exists	inv: definition <> Void
uid_validity	inv: uid <> Void implies not uid.is_empty
original_language_valid	inv: original_language <> Void and translations.language <> Void and terminology_service.code_set('languages').has(original_language)
has_parent	post: is_specialised implies parent_archetype_id <>Void

7.4 Пакет archetype_description

7.4.1 Общая информация

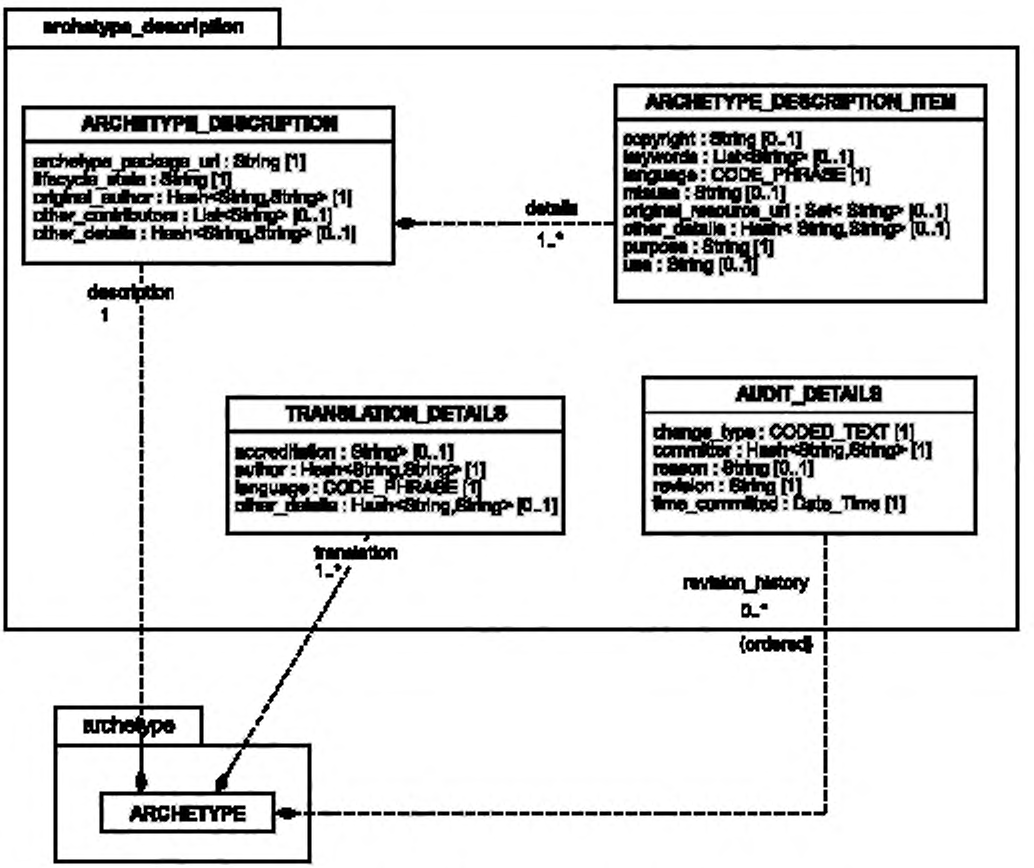

Рисунок 6 — Пакет archetype_description

7.4.2 Пакет :: archetype_description

«Метаданные» архетипа.

Внутренние элементы

Имя	Тип
ARCHETYPE_DESCRIPTION	Класс (class)
ARCHETYPE_DESCRIPTION_ITEM	Класс (class)
AUDIT_DETAILS	Класс (class)
TRANSLATION_DETAILS	Класс (class)

Пакет: archetype_description

Класс AUDIT_DETAILS

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
change_type : CODED_TEXT	1	- -	Тип изменения
committer : Hash<String,String>	1	- -	Подробная идентификация автора основного содержания архетипа, выраженная в виде списка пар «наименование-значение»
reason : String	0..1	- -	Причина изменения, выраженная на естественном языке
revision : String	1	- -	Версия, соответствующая данному изменению
time_committed : Date_Time	1	- -	Дата и время данного изменения

Ограничения

Имя	Выражение
committer_validity	inv: committer <> Void and not committer.is_empty
committer_organization_validity	inv: committer_organisation <> Void implies not committer_organisation.is_empty
time_committed_exists	inv: time_committed <> Void
reason_valid	inv: reason <> Void implies not reason.is_empty
revision_valid	inv: revision <> Void and not revision.is_empty
change_type_exists	inv: change_type <> Void and terminology_service.terminology('openehr').codes_for_group_name('audit_change_type', 'en').has (change_type.defining_code)

Пакет: archetype_description

Класс ARCHETYPE_DESCRIPTION

Данный класс определяет описательные метаданные архетипа.

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
archetype_package_uri : String	0..1	- -	Идентификатор URI пакета, к которому принадлежит данный архетип
lifecycle_state : String	1	- -	Состояние жизненного цикла архетипа: initial (начальный), submitted (представлен на рассмотрение), experimental (экспериментальный), awaiting_approval (ожидающий утверждения), approved (утвержден), superseded (заменен), obsolete (устарел)
original_author : Hash<String,String>	1	- -	Исходный автор данного архетипа, описанный в виде списка пар «имя-значение»
other_contributors : List<String>	0..1	- -	Имена других создателей данного архетипа
other_details : Hash<String,String>	0..1	- -	Дополнительные метаданные архетипа не на естественном языке, представленные в виде списка пар «имя-значение»

Атрибут, унаследованный от ассоциации

Сигнатура	Обязательность	Кратность	Описание
details : Set<ARCHETYPE_DESCRIPTION_ITEM>	1	1..*	Описательные метаданные архетипа

Ограничения

Имя	Выражение
original_author_validity	inv: original_author <> Void and not original_author.is_empty
details_exists	inv: details <> Void and not details.is_empty
original_author_organisation_validity	inv: original_author_organisation <> Void implies not original_author_organisation.is_empty
language_validity	inv: details->for_all(d| parent_archetype.languages_available.has(d.language))
parent_archetype_valid	inv: parent_archetype <> Void and parent_archetype.description = Current

Пакет: archetype_description

Класс ARCHETYPE_DESCRIPTION_ITEM

Детали описания архетипа, специфичные для языка. Если архетип переводится для использования в другой языковой среде, то каждый экземпляр класса ARCHETYPE_ DESCRIPTION_ITEM должен быть скопирован и переведен на новый язык.

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
copyright : String	0..1	- -	Необязательное предупреждение об авторском праве на архетип как источник знаний
keywords : List<String>	0..1	- -	Ключевые слова для поиска данного архетипа
language : CODE_PHRASE	1	- -	Местный язык, на котором написаны элементы данного описания
misuse : String	0..1	- -	Описание тех контекстов, в которых данный архетип не может использоваться
original_resource_uri : Set<String>	0..1	- -	Идентификатор URI исходного клинического документа (документов) или описания, формализацией которого является архетип, на языке данного элемента описания
other_details : Hash<String,String>	0..1	- -	Дополнительные метаданные архетипа, зависящие от языка и представленные в виде пар «имя-значение»
purpose : String	1	- -	Назначение данного архетипа
use : String	0..1	- -	Описание применения архетипа, т. е. контекстов, в которых он может использоваться

Ограничения

Имя	Выражение
use_valid	inv: use <> Void implies not use.is_empty
language_valid	inv: language <> Void and code_set('languages').has(language)
misuse_valid	inv: misuse <> Void implies not misuse.is_empty
copyright_valid	inv: copyright <> Void implies not copyright.is_empty
purpose_exists	inv: purpose <> Void and not purpose.is_empty

Пакет: archetype_description

Класс TRANSLATION_DETAILS

Класс, описывающий детали перевода на естественный язык.

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
accreditation : String	0..1	- -	Аккредитация переводчика, например, идентификатор национальной ассоциации переводчиков
author : Hash<String,String>	1	- -	Имя переводчика и другие демографические детали в виде пар «имя-значение»
language : CODE_PHRASE	1	- -	Язык перевода
other_details : Hash<String,String>	0..1	- -	Любые другие метаданные

7.5 Пакет constraint_model

7.5.1 Общая информация

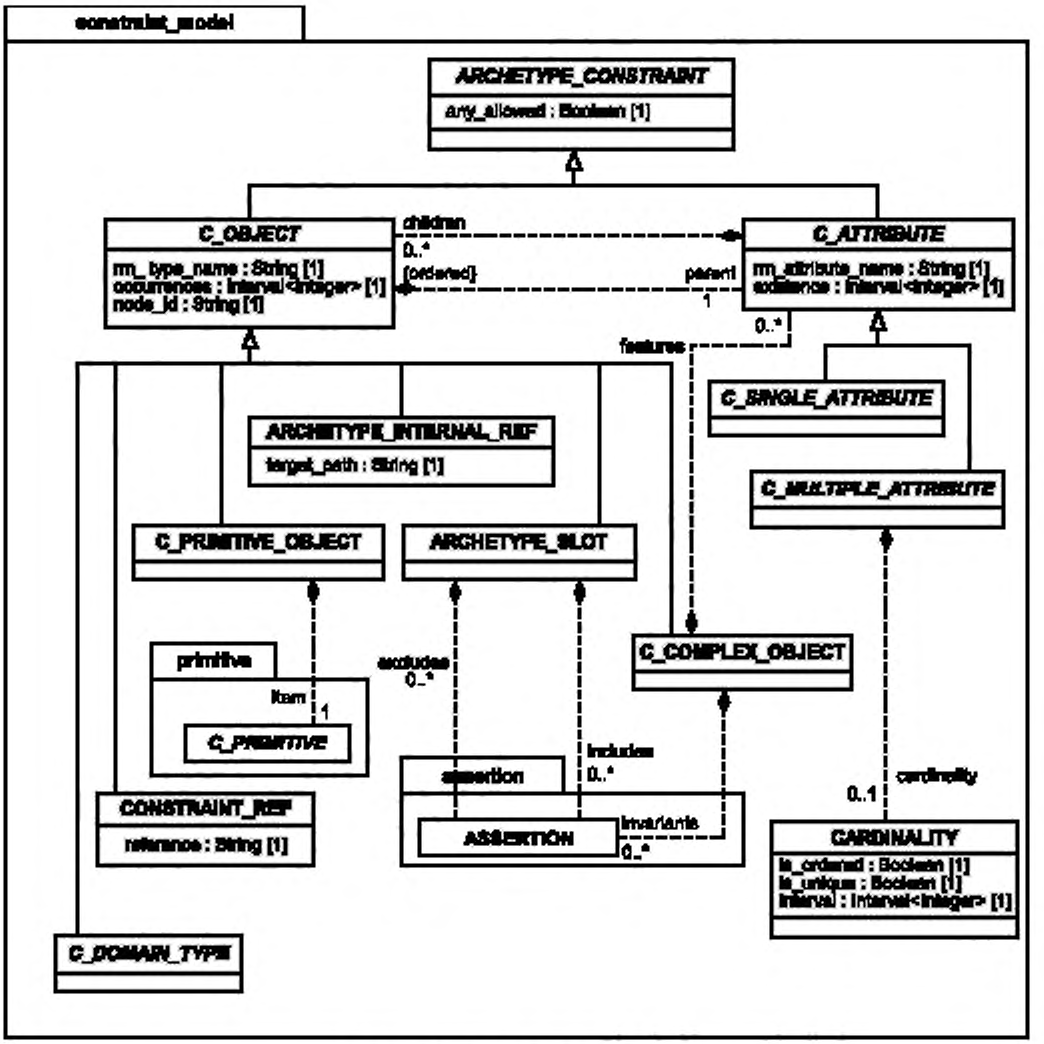

Рисунок 7 — Пакет constraint_model

7.5.2 Пакет :: constraint_model

Внутренние элементы

Имя	Выражение
ARCHETYPE_CONSTRAINT	Класс (class)
ARCHETYPE_INTERNAL_REF	Класс (class)
ARCHETYPE_SLOT	Класс (class)
C_ATTRIBUTE	Класс (class)
C_COMPLEX_OBJECT	Класс (class)

Окончание таблицы

Имя	Выражение
C_DOMAIN_TYPE	Класс (class)
C_MULTIPLE_ATTRIBUTE	Класс (class)
C_OBJECT	Класс (class)
C_PRIMITIVE_OBJECT	Класс (class)
C_SINGLE_ATTRIBUTE	Класс (class)
CARDINALITY	Класс (class)
CONSTRAINT_REF	Класс (class)
assertion	Пакет (package)
primitive	Пакет (package)

Пакет: constraint_model

Класс ARCHETYPE_CONSTRAINT{Abstract}

Непосредственные подклассы: C_OBJECT, C_ATTRIBUTE

Определяет общие ограничения любого класса архетипа в любой базовой модели.

Атрибут

Сигнатура	Обязательность	Кратность	Описание
any_allowed . Boolean	1	- -	Имеет значение «истина», если в архетипе не определены никакие дополнительные ограничения, кроме описанных в исходной базовой модели

Пакет: constraint_model

Класс C_ATTRIBUTE{Abstract}

ARCHETYPE_CONSTRAINT

|

+ - - C_ATTRIBUTE

Непосредственные подклассы: C_MULTIPLE_ATTRIBUTE, C_SINGLE_ATTRIBUTE

Абстрактная модель ограничения любого типа узла атрибута.

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
existence : Interval<Integer>	1	- -	Ограничение каждого атрибута, независимо от того, является ли он атрибутом одиночного или контейнерного типа. Указывает, существует ли его целевой объект (т. е. является он обязательным или нет)
rm_attribute_name : String	1	- -	Атрибут базовой модели в рамках объемлющего типа данных, представленный объектом C_OBJECT

Атрибут, унаследованный от ассоциации

Сигнатура	Обязательность	Кратность	Описание
children : List<C_OBJECT>	0..1	0..* ordered	Дочерние узлы C_OBJECT. Каждый такой узел описывает ограничение типа данного атрибута в его базовой модели

Ограничения

Имя	Выражение
existence_set	inv: existence <> Void and (existence.lower >= 0 and existence.upper <= 1)
rm_attribute_name_valid	inv: rm_attribute_name <> Void and not rm_attribute_name.is_empty
Children_validity	inv: any_allowed xor children <> Void

Пакет: constraint_model

Класс C_OBJECT{Abstract}

ARCHETYPE_CONSTRAINT

|

+ - - **C_OBJECT**

Непосредственные подклассы: ARCHETYPE_INTERNAL_REF, C_PRIMITIVE_OBJECT, C_COMPLEX_OBJECT, ARCHETYPE_SLOT, CONSTRAINT_REF, C_DOMAIN_TYPE.

Абстрактная модель ограничения любого типа узла объекта.

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
node_id : String	1	- -	Семантический идентификатор данного узла, используемый для различения сестринских узлов одного типа. (Ранее назывался «смыслом».) Каждый атрибут node_id должен быть определен в онтологии архетипа как код термина
occurrences : Interval<Integer>	1	- -	Вхождения данного узла объекта в данные под атрибутом-владельцем. Верхний предел кратности может быть только больше 1, если атрибут-владелец имеет кратность больше 1
rm_type_name : String	1	- -	Тип базовой модели, которому соответствует данный узел

Атрибуты, унаследованные от ассоциаций

Сигнатура	Обязательность	Кратность	Описание
parent : C_ATTRIBUTE	1	- -	Объект типа C_ATTRIBUTE, владеющий данным объектом типа C_OBJECT

Ограничения

Имя	Выражение
rm_type_name_valid	inv: rm_type_name <> Void and not rm_type_name.is_empty
node_id_valid	inv: node_id <> Void and not node_id.is_empty

Пакет: constraint_model

Класс CARDINALITY

Выражает ограничения кратности объектов контейнера, являющихся значениями многозначных атрибутов, включая уникальность и упорядочение. С их помощью можно указать, является ли контейнер логическим списком, множеством или пакетом. Эта кратность не может противоречить кратности соответствующего атрибута в релевантной базовой модели.

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
interval : Interval<Integer>	1	--	Интервал (диапазон) данной кратности
is_ordered : Boolean	1	--	Имеет значение «истина», если элементы контейнера атрибутов, к которому относится данная кратность, упорядочены
is_unique : Boolean	1	--	Имеет значение «истина», если элементы контейнера атрибутов, к которому относится данная кратность, уникальны

Ограничение

Имя	Выражение
Validity	inv: not interval.lower_unbounded

Пакет: constraint_model

Класс CONSTRAINT_REF

C_OBJECT

|

+ -- **CONSTRAINT_REF**

Ссылка на ограничение, описанное в том же архетипе, но вне основной структуры ограничений. Данный класс используется для ссылок на ограничения, выраженные в терминах внешних ресурсов, например ограничения совокупности терминологических значений.

Атрибут

Сигнатура	Обязательность	Кратность	Описание
reference : String	1	--	Ссылка на ограничение из локальной онтологии архетипа

Ограничения

Имя	Выражение
Consistency	inv: not any_allowed
reference_valid	inv: reference <> Void and not reference.is_empty and archetype.ontology.has_constraint(reference)

Пакет: constraint_model

Класс ARCHETYPE_INTERNAL_REF

C_OBJECT

|

+ -- **ARCHETYPE_INTERNAL_REF**

Ограничение, определенное посредником с использованием ссылки на ограничение объекта, определенное в другом месте того же архетипа.

Атрибут

Сигнатура	Обязательность	Кратность	Описание
target_path : String	1	--	Ссылка на узел объекта с использованием нотации пути архетипа

Ограничения

Имя	Выражение
Consistency	inv: not any_allowed
target_path_valid	inv: target_path <> Void and not target_path.is_empty and ultimate_root.has_path(target_path)

Пакет: constraint_model

Класс ARCHETYPE_SLOT

C_OBJECT

|

+ - - **ARCHETYPE_SLOT**

Ограничение, описывающее «слот», в котором может находиться другой архетип.

Атрибуты, унаследованные от ассоциаций

Сигнатура	Обязательность	Кратность	Описание
excludes : Set<ASSERTION>	0..1	0..*	Список ограничений, указывающий другие архетипы, которые не могут быть включены в данной точке
includes : Set<ASSERTION>	0..1	0..*	Список ограничений, указывающий другие архетипы, которые могут быть включены в данной точке

Ограничения

Имя	Выражение
includes_valid	inv: includes <> Void implies not includes.is_empty
excludes_valid	inv: excludes <> Void implies not excludes.is_empty
Validity	inv: any_allowed xor includes <> Void or excludes <> Void

Пакет: constraint_model

Класс C_SINGLE_ATTRIBUTE

C_ATTRIBUTE

|

+ - - **C_SINGLE_ATTRIBUTE**

Конкретная модель ограничения узла атрибута с единственным значением. Смысл унаследованных атрибутов-потомков в том, что они являются альтернативами.

Пакет: constraint_model

Класс C_MULTIPLE_ATTRIBUTE

C_ATTRIBUTE

|

+ - - **C_MULTIPLE_ATTRIBUTE**

Абстрактная модель ограничения любого узла атрибута.

Атрибут, унаследованный от ассоциации

Сигнатура	Обязательность	Кратность	Описание
cardinality : CARDINALITY	0..1	- -	Кратность данного ограничения атрибута, если оно ограничивает атрибут контейнерного типа

Ограничения

Имя	Выражение
members_valid	inv: members <> Void and members->for_all(co: C_OBJECT | co.occurrences.upper <= 1)
cardinality_validity	inv: cardinality <> Void

Пакет: constraint_model

Класс C_DOMAIN _TYPE{Abstract}

C_OBJECT

|

+ - - **C_DOMAIN_TYPE**

Непосредственные подклассы: C_ORDINAL, C_QUANTITY, C_CODED_TEXT.

Абстрактный родительский тип ограничивающих типов данных, специфичных для предметной области и определяемых во внешних пакетах.

Пакет: constraint_model

Класс C_COMPLEX_OBJECT

C_OBJECT

|

+ - - **C_COMPLEX_OBJECT**

Ограничение комплексных объектов, т. е. любой объект, состоящий из других ограничений объектов.

Атрибуты, унаследованные от ассоциаций

Сигнатура	Обязательность	Кратность	Описание
invariants : Set<ASSERTION>	0..1	0..*	Инвариантные утверждения о данном объекте. Утверждения выражаются с помощью логики предикатов первого порядка и обычно относятся, по крайней мере, к двум атрибутам
features : Set<C_ATTRIBUTE>	0..1	0..*	Список ограничений атрибутов базовой модели, представленных данным объектом

Ограничения

Имя	Выражение
attributes_valid	inv: any_allowed xor (attributes <> Void and not attributes.is_empty)
invariants_valid	inv: invariants <> Void implies not invariants.is_empty
invariant_consistency	inv: any_allowed implies invariants = Void

Пакет: constraint_model

Класс C_PRIMITIVE_OBJECT

C_OBJECT

|

+ - - **C_PRIMITIVE_OBJECT**

Атрибут, унаследованный от ассоциации

Сигнатура	Обязательность	Кратность	Описание
item : C_PRIMITIVE	1	- -	Объект, реально определяющий ограничение

Ограничение

Имя	Выражение
item_exists	inv: any_allowed xor item <> Void

7.6 Пакет assertion

7.6.1 Общая информация

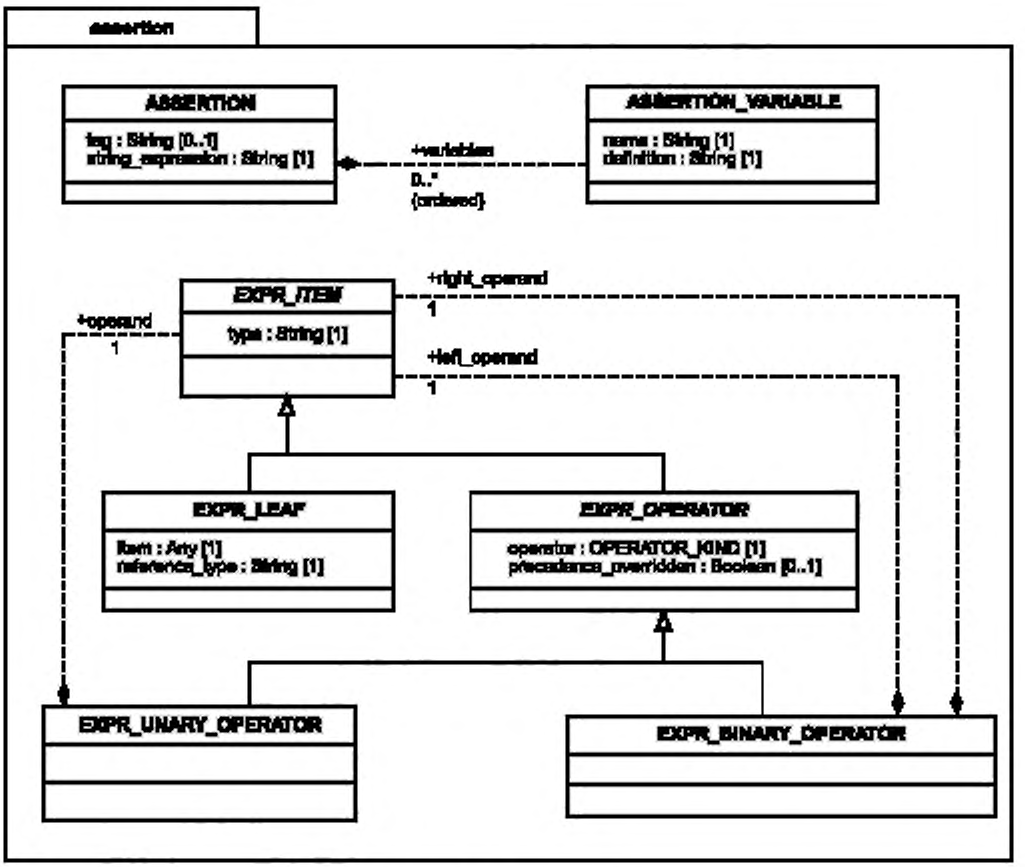

Рисунок 8 — Пакет assertion

7.6.2 Пакет :: assertion

Внутренние элементы

Имя	Тип
ASSERTION	Класс (class)
ASSERTION_VARIABLE	Класс (class)
EXPR_BINARY_OPERATOR	Класс (class)
EXPR_ITEM	Класс (class)
EXPR_LEAF	Класс (class)
EXPR_OPERATOR	Класс (class)
EXPR_UNARY_OPERATOR	Класс (class)

Пакет: assertion

Класс EXPR_ITEM{Abstract}

Непосредственные подклассы: EXPR_OPERATOR, EXPR_LEAF.

Атрибут

Сигнатура	Обязательность	Кратность	Описание
type : String	1	--	Отсутствует

Ограничение

Наименование	Выражение
type_valid	inv: type <> Void and not type.is_empty

Пакет: assertion

Класс EXPR_LEAF

EXPR_ITEM

|

+ - - **EXPR_LEAF**

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
item : Any	1	--	Отсутствует
reference_type : String	1	--	Отсутствует

Ограничение

Наименование	Выражение
item_valid	inv: item <> Void

Пакет: assertion

Класс EXPR_TOR{Abstract}

EXPR_ITEM

|

+ - - **EXPR_OPERATOR**

Непосредственные подклассы: EXPR_BINARY_OPERATOR, EXPR_UNARY_OPERATOR.

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
operator : OPERATOR_KIND	1	--	Отсутствует
precedence_overridden : Boolean	1	--	Отсутствует

Пакет: assertion

Класс EXPR_UNARY_OPERATOR

EXPR_OPERATOR

|

+ - - **EXPR_UNARY_OPERATOR**

Атрибут, унаследованный от ассоциации

Сигнатура	Обязательность	Кратность	Описание
operand : EXPR_ITEM	1	--	Отсутствует

Ограничение

Имя	Выражение
operand_valid	inv: operand <> Void

Пакет: assertion

Класс EXPR_BINARY_OPERATOR

EXPR_OPERATOR

|

+ - - **EXPR_BINARY_OPERATOR**

Атрибуты, унаследованные от ассоциаций

Сигнатура	Обязательность	Кратность	Описание
left_operand : EXPR_ITEM	1	- -	Отсутствует
right_operand : EXPR_ITEM	1	- -	Отсутствует

Ограничения

Имя	Выражение
left_operand_valid	inv: left_operand <> Void
right_operand_valid	inv: right_operand <> Void

Пакет: assertion

Класс ASSERTION_VARIABLE

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
definition : String	1	- -	Отсутствует
Name : String	1	- -	Отсутствует

Пакет: assertion

Класс ASSERTION

Структурная модель типизированного утверждения, представленного в формализме логики предикатов первого порядка в виде дерева выражений, включая необязательные определения переменных.

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
string_expression : String	1	- -	Отсутствует
tag : String	0..1	- -	Отсутствует

Атрибуты, унаследованные от ассоциаций

Сигнатура	Обязательность	Кратность	Описание
variables : List<ASSERTION_VARIABLE>	0..1	0..* ordered	Отсутствует
expression : EXPR_ITEM	1	- -	Отсутствует

Ограничения

Имя	Выражение
expression_valid	inv: expression <> Void and expression.type.is_equal(«Boolean»)
tag_valid	inv: tag <> Void implies not tag.is_empty

7.7 Пакет primitive

7.7.1 Общая информация

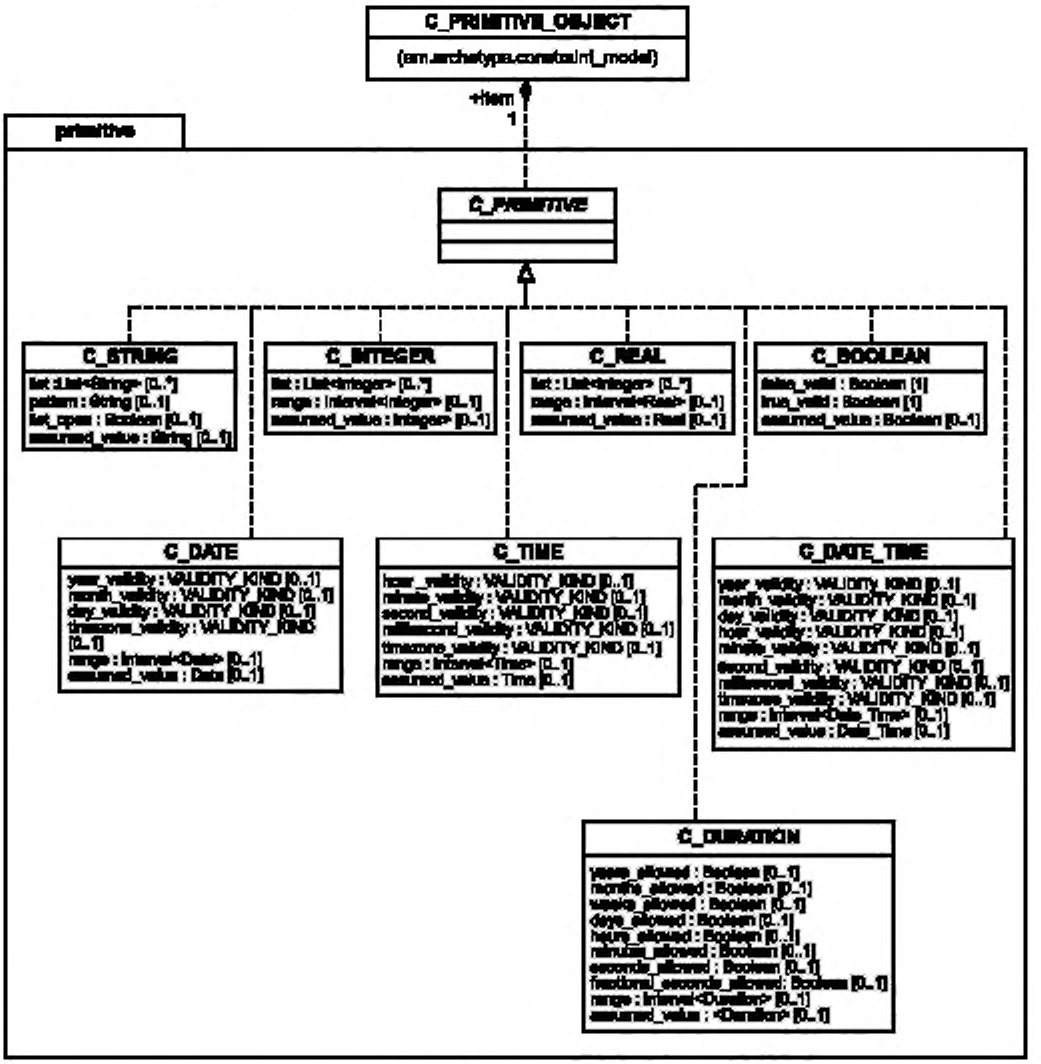

Рисунок 9 — Пакет primitive

7.7.2 Пакет :: primitive

Внутренние элементы

Имя	Тип
C_BOOLEAN	Класс (class)
C_DATE	Класс (class)
C_DATE_TIME	Класс (class)
C_DURATION	Класс (class)

Окончание таблицы

Имя	Тип
C_INTEGER	Класс (class)
C_PRIMITIVE	Класс (class)
C_REAL	Класс (class)
C_STRING	Класс (class)
C_TIME	Класс (class)

Пакет: primitive

Класс C_STRING

C_PRIMITIVE

|

+ - - **C_STRING**

Ограничение экземпляров типа String (строка).

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
assumed_value : String	0..1	- -	Предполагаемое значение на тот случай, если элемент является частью необязательной структуры и не включен в данные
list : Set<List<String>>	0..1	0..*	Список элементов типа String, описывающих ограничение
list_open : Boolean	0..1	- -	Имеет значение «истина», если список используется для описания ограничения, но не является исчерпывающим
pattern : String	0..1	- -	Шаблон регулярного выражения, которому должны удовлетворять предложенные экземпляры типа String

Ограничения

Имя	Выражение
pattern_exists	inv: pattern <> Void implies not pattern.is_empty
Consistency	inv: pattern <> Void xor list <> Void

Пакет: primitive

Класс C_BOOLEAN

C_PRIMITIVE

|

+ - - **C_BOOLEAN**

Ограничение экземпляров типа Boolean (булевский). Оба атрибута не могут одновременно принимать значение «ложь», так как это бы означало, что ограничиваемое булевское значение не может иметь значение ни «истина», ни «ложь».

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
assumed_value : Boolean	0..1	- -	Предполагаемое значение на тот случай, если элемент является частью необязательной структуры и не включен в данные
false_valid : Boolean	1	- -	Имеет значение «истина», если допустимо значение «ложь»
true_valid : Boolean	1	- -	Имеет значение «истина», если допустимо значение «истина»

Ограничения

Имя	Выражение
Default_value_consistency	inv: (default_value.value and true_valid) or (not default_value.value and false_valid)
Binary_consistency	inv: true_valid or false_valid

Пакет: primitive

Класс C_DURATION

C_PRIMITIVE

|

+ - - **C_DURATION**

Ограничение экземпляров типа Duration (длительность).

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
assumed_value : Duration	0..1	- -	Предполагаемое значение на тот случай, если элемент является частью необязательной структуры и не включен в данные
days_allowed : Boolean	0..1	- -	Имеет значение «истина», если в ограниченном значении типа Duration допустимо указание дней
fractional_seconds_allowed : Boolean	0..1	- -	Имеет значение «истина», если в ограниченном значении типа Duration допустимы доли секунд
hours_allowed : Boolean	0..1	- -	Имеет значение «истина», если в ограниченном значении типа Duration допустимы часы
minutes_allowed : Boolean	0..1	- -	Имеет значение «истина», если в ограниченном значении типа Duration допустимы минуты
months_allowed : Boolean	0..1	- -	Имеет значение «истина», если в ограниченном значении типа Duration допустимы месяцы
range : Interval<Duration>	0..1	- -	Ограничение экземпляров типа Duration
seconds_allowed : Boolean	0..1	- -	Имеет значение «истина», если в ограниченном значении типа Duration допустимы секунды
weeks_allowed : Boolean	0..1	- -	Имеет значение «истина», если в ограниченном значении типа Duration допустимы недели
years_allowed : Boolean	0..1	- -	Имеет значение «истина», если в ограниченном значении типа Duration допустимы годы

Ограничение

Имя	Выражение
Range_valid	inv: range <> Void

Пакет: primitive
Класс C_DATE_TIME
C_PRIMITIVE
|
+ - - **C_DATE_TIME**

Ограничение экземпляров типа Date_Time (дата и время). Флаг наличия объекта «year (год)» не существует, так как по определению этот объект всегда обязателен, иначе дата и время будут лишены смысла.

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
assumed_value : Date_Time	0..1	- -	Предполагаемое значение на тот случай, если элемент является частью необязательной структуры и не включен в данные
day_validity : VALIDITY_KIND	0..1	- -	Допустимость наличия дня в ограниченной дате
hour_validity : VALIDITY_KIND	0..1	- -	Допустимость наличия часов в ограниченном времени
millisecond_validity : VALIDITY_KIND	0..1	- -	Допустимость наличия миллисекунд в ограниченном времени
minute_validity : VALIDITY_KIND	0..1	- -	Допустимость наличия минут в ограниченном времени
month_validity : VALIDITY_KIND	0..1	- -	Допустимость наличия месяца в ограниченной дате
range : Interval<Date_Time>	0..1	- -	Диапазон значений типа Date_Time, задающий ограничение
second_validity : VALIDITY_KIND	0..1	- -	Допустимость наличия секунд в ограниченном времени
timezone_validity : VALIDITY_KIND	0..1	- -	Допустимость наличия часового пояса в ограниченной дате
year_validity : VALIDITY_KIND	0..1	- -	Отсутствует

Ограничения

Имя	Выражение
second_validity_disallowed	inv: second_validity = 'disallowed' implies millisecond_validity = 'disallowed'
second_validity_optional	inv: second_validity = 'optional' implies (millisecond_validity = 'optional' or millisecond_validity = 'disallowed')
minute_validity_optional	inv: minute_validity = 'optional' implies (second_validity = 'optional' or second_validity = 'disallowed')
minute_validity_disallowed	inv: minute_validity = 'disallowed' implies second_validity = 'disallowed'
hour_validity_disallowed	inv: hour_validity = 'disallowed' implies minute_validity = 'disallowed'
day_validity_disallowed	inv: day_validity = 'disallowed' implies hour_validity = 'disallowed'
month_validity_disallowed	inv: month_validity = 'disallowed' implies day_validity = 'disallowed'
day_validity_optional	inv: day_validity = 'optional' implies (hour_validity = 'optional' or hour_validity = 'disallowed')
hour_validity_optional	inv: hour_validity = 'optional' implies (minute_validity = 'optional' or minute_validity = 'disallowed')
validity_is_range	inv: validity_is_range = (range <> Void)

Пакет: primitive

Класс C_INTEGER

C_PRIMITIVE

|

+ - - C_INTEGER

Ограничение экземпляров типа Integer (целочисленное значение).

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
assumed_value : Integer	0..1	- -	Предполагаемое значение на тот случай, если элемент является частью необязательной структуры и не включен в данные
list : Set<List<Integer>>	0..1	0..*	Множество значений типа Integer, определяющее ограничение
range : Interval<Integer>	0..1	- -	Диапазон значений типа Integer, определяющий ограничение

Ограничение

Имя	Выражение
consistency	inv: list <> Void xor range <> Void

Пакет: primitive

Класс C_TIME

C_PRIMITIVE

|

+ - - C_TIME

Ограничение экземпляров типа Time (время). Флаг наличия объекта «hour (час)» не существует, так как по определению этот объект всегда обязателен, иначе время будет лишено смысла.

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
assumed_value : Time	0..1	- -	Предполагаемое значение на тот случай, если элемент является частью необязательной структуры и не включен в данные
hour_validity : VALIDITY_KIND	0..1	- -	Отсутствует
millisecond_validity : VALIDITY_KIND	0..1	- -	Допустимость наличия миллисекунд в ограниченном времени
minute_validity : VALIDITY_KIND	0..1	- -	Допустимость наличия минут в ограниченном времени
range : Interval<Date_Time>	0..1	- -	Диапазон значений типа Time, задающий ограничение
second_validity : VALIDITY_KIND	0..1	- -	Допустимость наличия секунд в ограниченном времени
timezone_validity : VALIDITY_KIND	0..1	- -	Допустимость наличия часового пояса в ограниченной дате

Ограничения

Имя	Выражение
minute_validity_optional	inv: minute_validity = 'optional' implies (second_validity = 'optional' or second_validity = 'disallowed')
second_validity_disallowed	inv: second_validity = 'disallowed' implies millisecond_validity = 'disallowed'
second_validity_optional	inv: second_validity = 'optional' implies (millisecond_validity = 'optional' or millisecond_validity = 'disallowed')
minute_validity_disallowed	inv: minute_validity = 'disallowed' implies second_validity = 'disallowed'
validity_is_range	inv: validity_is_range = (range <> Void)

Пакет: primitive

Класс C_REAL

C_PRIMITIVE

|

+ - - **C_REAL**

Ограничение экземпляров типа Real (действительное значение).

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
assumed_value : Real	0..1	- -	Предполагаемое значение на тот случай, если элемент является частью необязательной структуры и не включен в данные
list : Set<List<Real>>	0..1	0..*	Множество значений типа Real, определяющее ограничение
range : Interval<Real >	0..1	- -	Диапазон значений типа Real, определяющий ограничение

Ограничение

Имя	Выражение
consistency	inv: list <> Void xor range <> Void

Пакет: primitive

Класс C_DATE

C_PRIMITIVE

|

+ - - **C_DATE**

Ограничение экземпляров типа Date (дата) в виде допустимости наличия значений или диапазона реальных дат. Флаг наличия объекта «year (год)» не существует, так как по определению этот объект всегда обязателен, иначе дата будет лишена смысла.

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
assumed_value : Date	0..1	- -	Предполагаемое значение на тот случай, если элемент является частью необязательной структуры и не включен в данные
day_validity : VALIDITY_KIND	0..1	- -	Допустимость дней в ограниченной дате
month_validity : VALIDITY_KIND	0..1	- -	Допустимость наличия дня в ограниченной дате
range : Interval<Date>	0..1	- -	Диапазон значений типа Date, задающий ограничение
timezone_validity : VALIDITY_KIND	0..1	- -	Допустимость наличия часового пояса в ограниченной дате
year_validity : VALIDITY_KIND	0..1	- -	Отсутствует

Ограничения

Имя	Выражение
validity_is_range	inv: validity_is_range = (range <> Void)
month_validity_optional	inv: month_validity = 'optional' implies (day_validity = 'optional' or day_validity = 'disallowed')
month_validity_disallowed	inv: month_validity = 'disallowed' implies day_validity = 'disallowed'

Пакет: primitive

Класс C_PRIMITIVE{Abstract}

Непосредственные подклассы: C_REAL, C_BOOLEAN, C_STRING, C_DATE, C_DURATION, C_INTEGER, C_TIME, C_DATE_TIME.

7.8 Пакет ontology

7.8.1 Общая информация

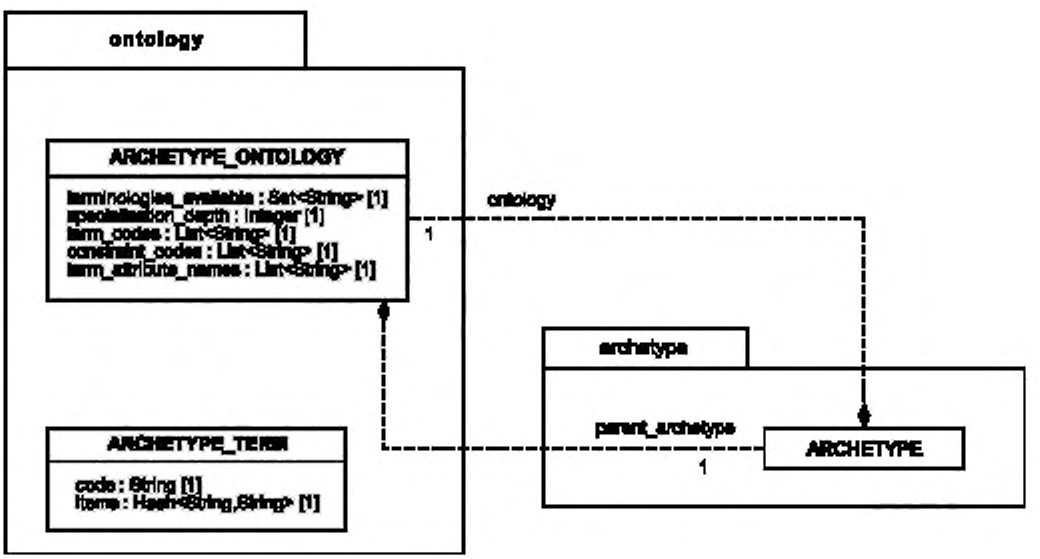

Рисунок 10 — Пакет ontology

7.8.2 Пакет :: ontology

Внутренние элементы

Имя	Тип
ARCHETYPE_ONTOLOGY	Класс (class)
ARCHETYPE_TERM	Класс (class)

Пакет: ontology

Класс ARCHETYPE_ONTOLOGY

Местная онтология архетипа.

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
constraint_codes : List<String>	1	--	Список всех кодов ограничений онтологии
specialisation_depth : Integer	1	--	Глубина специализации данного архетипа. Неспециализированные архетипы имеют глубину 0, и каждый дополнительный уровень специализации добавляет 1 к значению specialisation_depth
term_attribute_names : List<String>	1	--	Список имен "атрибутов" в онтологии терминов. Обычно включает "text" (текст), "description" (описание), "provenance" (происхождение) и т. д.
term_codes : List<String>	1	--	Список всех кодов терминов в онтологии. Большинство из них соответствует кодам "at" в описании архетипа на языке ADL, которые являются идентификаторами узлов node_id у потомков класса C_OBJECT. Кроме них, может существовать еще один дополнительный код, если в качестве общего понятия архетипа (объект concept_code) используется не идентификатор node_id самого верхнего объекта C_OBJECT в разделе определения архетипа, а другой термин
terminologies_available : Set<String>	1	--	Список терминологий, связи терминов или ограничений с которыми существуют в данной терминологии

Атрибут, унаследованный от ассоциаций

Сигнатура	Обязательность	Кратность	Описание
parent_archetype : ARCHETYPE	1	--	Архетип, владеющий данной онтологией

Ограничения

Имя	Выражение
terminologies_available_exists	inv: terminologies_available <> Void
term_attribute_names_valid	inv: term_attribute_names <> Void and term_attribute_names.has('text') and term_attribute_names.has('description')
Parent_archetype_valid	inv: parent_archetype <> Void and parent_archetype.description = Current
constraint_codes_exists	inv: constraint_codes <> Void
concept_code_valid	inv: term_codes.has (concept_code)
term_codes_exists	inv: term_codes <> Void

Пакет: ontology

Класс ARCHETYPE_TERM

Представление любого кодированного объекта (термина или ограничения) в онтологии архетипа.

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
code : String	1	--	Код данного термина
items : Hash<String,String>		--	Хэш-преобразование ключей ("text", "description" и т. д.) и соответствующих им значений

Ограничение

Имя	Выражение
code_valid	inv: code <> Void and not code.is_empty

7.9 Пакет domain_extensions

7.9.1 Общая информация

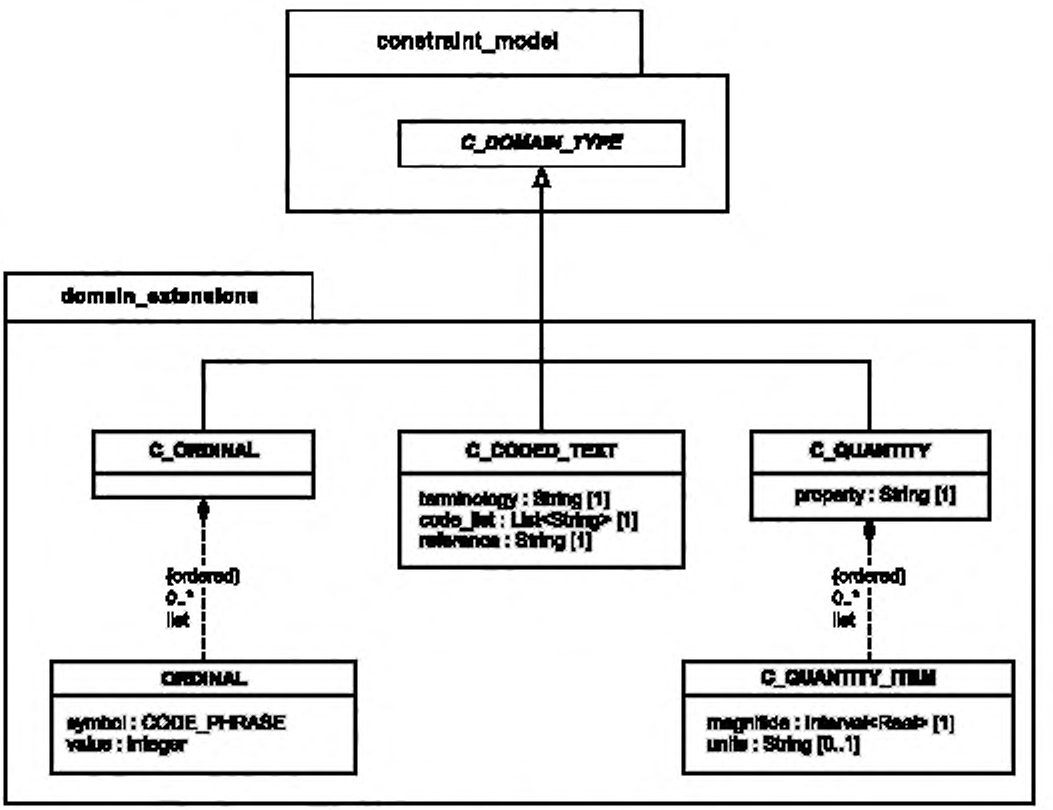

Рисунок 11 — Пакет domain_extensions

7.9.2 Пакет :: domain_extensions

Внутренние элементы

Имя	Тип
C_CODED_TEXT	Класс (class)
C_ORDINAL	Класс (class)
C_QUANTITY	Класс (class)
C_QUANTITY_ITEM	Класс (class)
ORDINAL	Класс (class)

Пакет: domain_extensions

Класс C_ORDINAL

C_DOMAIN_TYPE

|

+ - - **C_ORDINAL**

Атрибут, унаследованный от ассоциации

Сигнатура	Обязательность	Кратность	Описание
list : List<ORDINAL>	0..1	0..* ordered	Отсутствует

Пакет: domain_extensions

Класс C_CODED_TEXT

C_DOMAIN_TYPE

|

+ - - **C_CODED_TEXT**

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
code_list . List<String>	1	- -	Отсутствует
reference : String	1	- -	Отсутствует
terminology : String	1	- -	Отсутствует

Пакет: domain_extensions
Класс C_QUANTITY
C_DOMAIN_TYPE
|
+ - - **C_QUANTITY**
Атрибут

Сигнатура	Обязательность	Кратность	Описание
property : String	1	- -	Отсутствует

Атрибут, унаследованный от ассоциации

Сигнатура	Обязательность	Кратность	Описание
list : List<C_QUANTITY_ITEM >	0..1	0..* ordered	Отсутствует

Пакет: domain_extensions
Класс C_QUANTITY_ITEM
Атрибуты

Сигнатура	Обязательность	Кратность	Описание
magnitude : Interval<Real>	1	- -	Отсутствует
units : String	0..1	- -	Отсутствует

Пакет: domain_extensions
Класс ORDINAL
Атрибуты

Сигнатура	Обязательность	Кратность	Описание
symbol : CODE_PHRASE	0..1	- -	Отсутствует
value : Integer	0..1	- -	Отсутствует

7.10 Пакет support

7.10.1 Общая информация

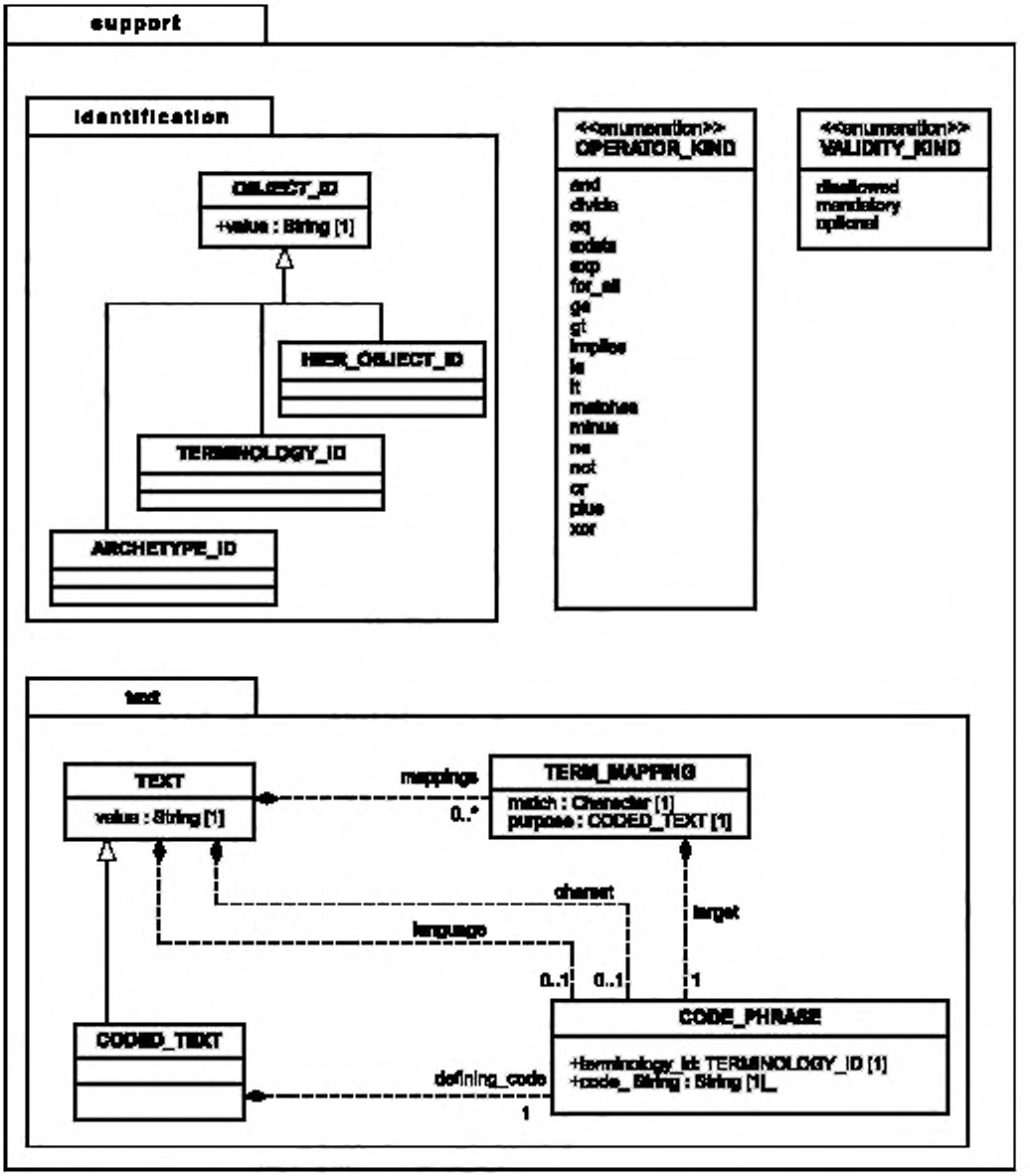

Рисунок 12 — Пакет support

7.10.2 Пакет :: support

Внутренние элементы

Имя	Тип
OPERATOR_KIND	Перечисление (enumeration)
VALIDITY_KIND	Перечисление (enumeration)
Identification	Пакет (package)
text	Пакет (package)

Пакет: support

Перечисление OPERATOR_KIND

Литералы перечисления:

- and (и);
- divide (делить);
- eq (равно);
- exists (существует);
- exp (экспонента);
- for_all (для всех);
- ge (больше или равно);
- gt (больше);
- implies (влечет за собой);
- le (меньше или равно);
- lt (меньше);
- matches (совпадает при сравнении);
- minus (минус);
- ne (не равно);
- not (не);
- or (или);
- plus (плюс);
- xor (исключающее или).

Пакет: support

Перечисление VALIDITY_KIND

Литералы перечисления:

- disallowed (запрещенный);
- mandatory (обязательный);
- optional (необязательный).

7.10.3 Пакет :: Identification

Внутренние элементы

Имя	Тип
ARCHETYPE_ID	Класс (class)
HIER_OBJECT_ID	Класс (class)
OBJECT_ID	Класс (class)
TERMINOLOGY_ID	Класс (class)

Пакет: Identification

Класс TERMINOLOGY_ID

OBJECT_ID

|

+ - - **TERMINOLOGY_ID**

Пакет: Identification

Класс OBJECT_ID{Abstract}

Непосредственные подклассы: HIER_OBJECT_ID, TERMINOLOGY_ID, ARCHETYPE_ID.

Атрибут

Сигнатура	Обязательность	Кратность	Описание
value : String	1	- -	Отсутствует

Пакет: Identification

Класс HIER_OBJECT_ID

OBJECT_ID

|

+ - - **HIER_OBJECT_ID**

Пакет: Identification

Класс ARCHETYPE_ID

OBJECT_ID

|

+ - - **ARCHETYPE_ID**

7.10.4 Пакет :: text

Внутренние элементы

Имя	Тип
CODE_PHRASE	Класс (class)
CODED_TEXT	Класс (class)
TERM_MAPPING	Класс (class)
TEXT	Класс (class)

Пакет: text

Класс CODED_TEXT

TEXT

|

+ - - **CODED_TEXT**

Текстовый элемент, значением которого должна быть рубрика из контролируемой терминологии, ключом (т. е. «кодом») которой является атрибут defining_code. Другими словами: объект CODED_TEXT является комбинацией объекта CODE_PHRASE (собственно кода) и рубрики этого термина в терминологической службе на языке, на котором созданы данные.

Атрибут, унаследованный от ассоциации

Сигнатура	Обязательность	Кратность	Описание
defining_code : CODE_PHRASE	1	- -	Отсутствует

Пакет: text

Класс TEXT

Непосредственный подкласс: CODED_TEXT.

Элемент свободного текста, который может содержать любое число допустимых символов, сгруппированных в слова, предложения и т. д. (то есть один элемент класса TEXT может представлять более одного слова). Любой объект TEXT может быть «закодирован» с помощью добавления отображений на него.

Атрибут

Сигнатура	Обязательность	Кратность	Описание
value : String	1	--	Отображаемое представление элемента, не зависящее от лежащей в его основе структуры. Для объекта CODED_TEXT это рубрика полного термина в том виде, как она предоставляется терминологической службой. Без символов «возврат каретки», «перевод строки» или других непечатаемых символов

Атрибуты, унаследованные от ассоциаций

Сигнатура	Обязательность	Кратность	Описание
charset : CODE_PHRASE	0..1	--	Наименование набора символов, в котором закодировано данное значение
language : CODE_PHRASE	0..1	--	Необязательный индикатор местного языка, на котором написано данное значение
mappings : Set<TERM_MAPPING>	0..1	0..*	Термины из других терминологий, наиболее близко соответствующие данному термину

Пакет: text

Класс TERM_MAPPING

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
match : Character	1	--	Относительное соответствие целевого термина отображаемому текстовому элементу. Значение результата соответствия: «>»: отображение на более широкий термин; «=»: отображение (предположительно) эквивалентно исходному термину; «<»: отображение на более узкий термин; «?»: соответствие отображения неизвестно
purpose : CODED_TEXT	1	--	Цель отображения, например: «автоматический поиск данных», «выставление счета», «интероперабельность»

Атрибут, унаследованный от ассоциации

Сигнатура	Обязательность	Кратность	Описание
target: CODE_PHRASE	1	--	Целевой термин отображения

Пакет: text

Класс CODE_PHRASE

Полностью координированный (т. е. все действия «координирования» были выполнены) термин из терминологической службы (отличной от конкретной терминологии).

Атрибуты

Сигнатура	Обязательность	Кратность	Описание
code_string : String	1	--	Ключ, используемый терминологической службой для идентификации понятия или координирования понятий
terminology_id : TERMINOLOGY_ID	1	--	Идентификатор другой терминологии, из которой взят объект code_string (или его элементы)

7.10.5 Пакет :: generic_types

Внутренние элементы

Имя	Тип
Aggregate	Класс (class)
Hash	Класс (class)
Interval	Класс (class)
List	Класс (class)
Set	Класс (class)

Пакет: generic_types
Класс List
Aggregate
|
+ - - **LIST**
Атрибут

Сигнатура	Обязательность	Кратность	Описание
content : List<T>	0..1	0..* ordered	Отсутствует

Параметр шаблона

Имя	Тип	Значение по умолчанию
T	Отсутствует	Отсутствует

Пакет: generic_types
Класс Set
Aggregate
|
+ - - **Set**
Атрибут

Сигнатура	Обязательность	Кратность	Описание
content : Set<T>	0..1	0..*	Отсутствует

Параметр шаблона

Имя	Тип	Значение по умолчанию
T	Отсутствует	Отсутствует

Пакет: generic_types
Класс Interval
Атрибуты

Сигнатура	Обязательность	Кратность	Описание
lower : T	1	- -	Отсутствует
lower_unbounded : Boolean	1	- -	Отсутствует
upper : T	1	- -	Отсутствует
upper_unbounded : Boolean	1	- -	Отсутствует

Параметр шаблона

Имя	Тип	Значение по умолчанию
T	Отсутствует	Отсутствует

Пакет: generic_types
Класс Hash
Aggregate
|
+ - - **Hash**
Атрибуты

Сигнатура	Обязательность	Кратность	Описание
key : S	1	- -	Отсутствует
value : T	1	- -	Отсутствует

Параметр шаблона

Имя	Тип	Значение по умолчанию
S	Отсутствует	Отсутствует
T	Отсутствует	Отсутствует

Пакет: generic_types
Класс Aggregate{Abstract}
Непосредственные подклассы: List, Hash, Set.
Параметр шаблона

Имя	Тип	Значение по умолчанию
T	Отсутствует	Отсутствует

Операции

Сигнатура	Ограничение	Описание
count() : Integer	Отсутствует	Отсутствует
has() : Boolean	Отсутствует	Отсутствует
is_empty() : Boolean	Отсутствует	Отсутствует

7.10.6 Пакет :: primitive_data_types

Внутренние элементы

Имя	Тип
Any (любой)	Тип данных (Data Type)
Boolean (булевский)	Тип данных (Data Type)
Character (символ)	Тип данных (Data Type)
Date (дата)	Тип данных (Data Type)
Date_Time (дата и время)	Тип данных (Data Type)
Double (действительный двойной точности)	Тип данных (Data Type)
Duration (длительность)	Тип данных (Data Type)
Integer (целочисленный)	Тип данных (Data Type)
Real (действительный)	Тип данных (Data Type)
String (строковый)	Тип данных (Data Type)
Time (время)	Тип данных (Data Type)

Пакет: primitive_data_types
Тип данных Double
Any
|
+ - - **Double**
Пакет: primitive_data_types
Тип данных Character
Any
|
+ - - **Character**
Пакет: primitive_data_types
Тип данных Date
Any
|
+ - - **Date**
Пакет: primitive_data_types
Тип данных Duration
Any
|
+ - - **Duration**

Пакет: primitive_data_types
Тип данных Date_Time
Any
|
+ - - **Date_Time**
Пакет: primitive_data_types
Тип данных Time
Any
|
+ - - **Time**
Пакет: primitive_data_types
Тип данных String
Any
|
+ - - **String**

Операции

Сигнатура	Ограничение	Значение по умолчанию
is_empty() : Boolean	Отсутствует	Отсутствует
is_equal(other : String) : Boolean	Отсутствует	Отсутствует

Пакет: primitive_data_types
Тип данных Integer
Any
|
+ - - **Integer**
Пакет: primitive_data_types
Тип данных Boolean
Any
|
+ - - **Boolean**
Пакет: primitive_data_types
Тип данных Real
Any
|
+ - - **Real**
Пакет: primitive_data_types
Тип данных Any

Непосредственные подклассы: Date_Time, Real, Time, Integer, Duration, Character, String, Double, Boolean, Date.

7.11 Пакет generic_types

Данный пакет включен для подтверждения семантики обобщенных типов данных, используемых в настоящем стандарте. Хотя типы List<T>, Set<T>, Bag (не используется), Hash<T,K> и Interval<T> являются обобщенными типами данных, поддерживаемыми многими программными средами, они не поддерживаются в языке UML непосредственно. В данном пакете новые типы данных, например List<String> (список строк), определяются с использованием зависимостей связей между новым базовым типом (в данном случае List<String>) и классом (LIST в данном примере), который определяет минимальную необходимую семантику для всех типов данных List.

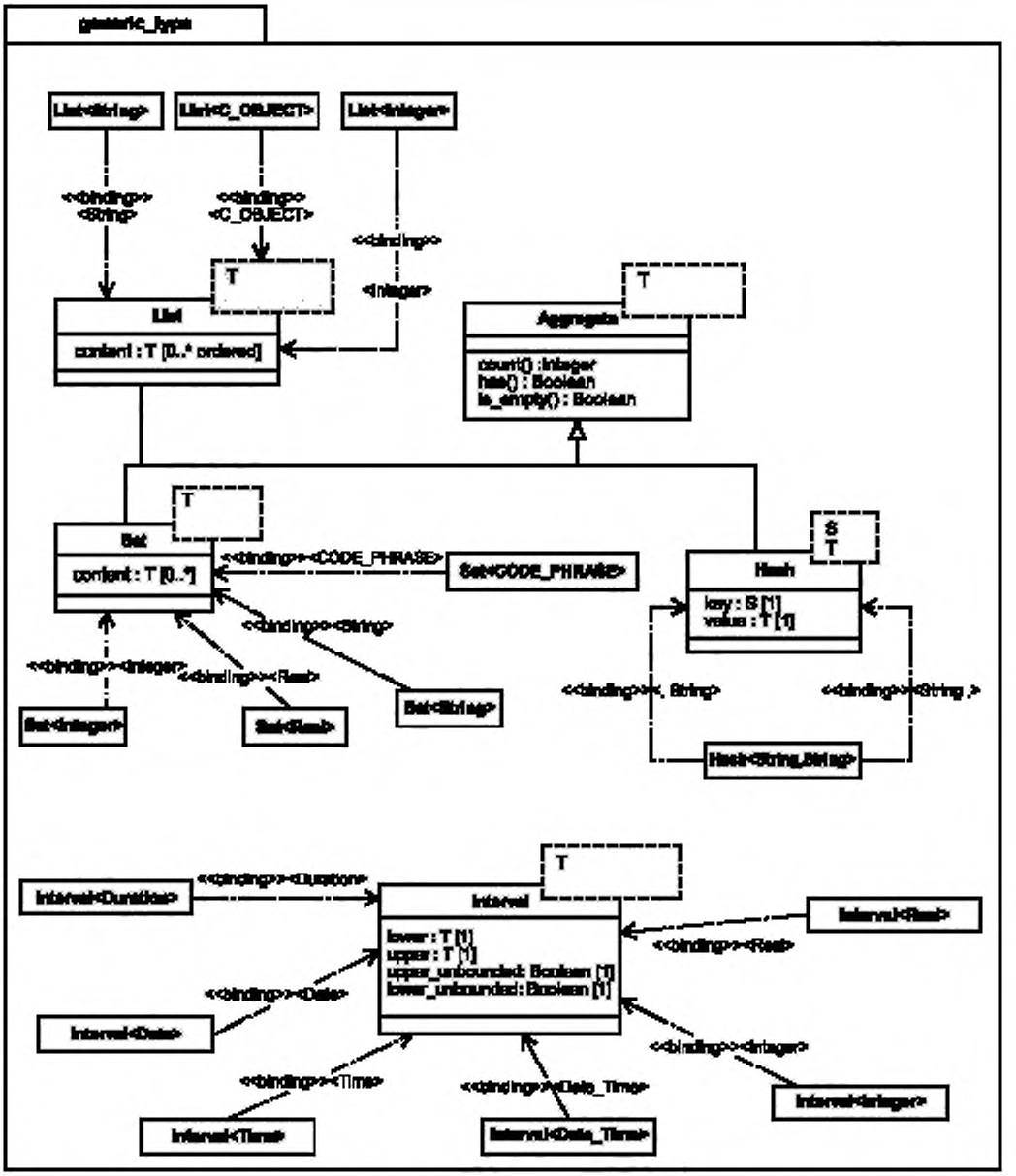

Рисунок 13 — Пакет generic_types

7.12 Расширения, специфичные для предметной области (справочно)

7.12.1 Общая информация

Специфичные для предметной области классы могут быть добавлены к модели ограничений архетипа с помощью наследования от класса *C_DOMAIN_TYPE*.

7.12.2 Научные/клинические вычислительные типы данных

На рисунке 14 показан общий подход, используемый для добавления классов ограничений повсеместно используемых понятий научных и клинических вычислений, например, «ordinal (порядковое числительное)», «coded term (кодированный термин)» и «quantity (количество)». На рисунке 14 показаны типы ограничений *C_ORDINAL*, *C_CODED* и *C_QUANTITY*, которые могут факультативно использоваться в архетипах вместо используемой по умолчанию семантики ограничений, представленной с помощью экземпляров классов *C_OBJECT* / *C_ATTRIBUTE*.

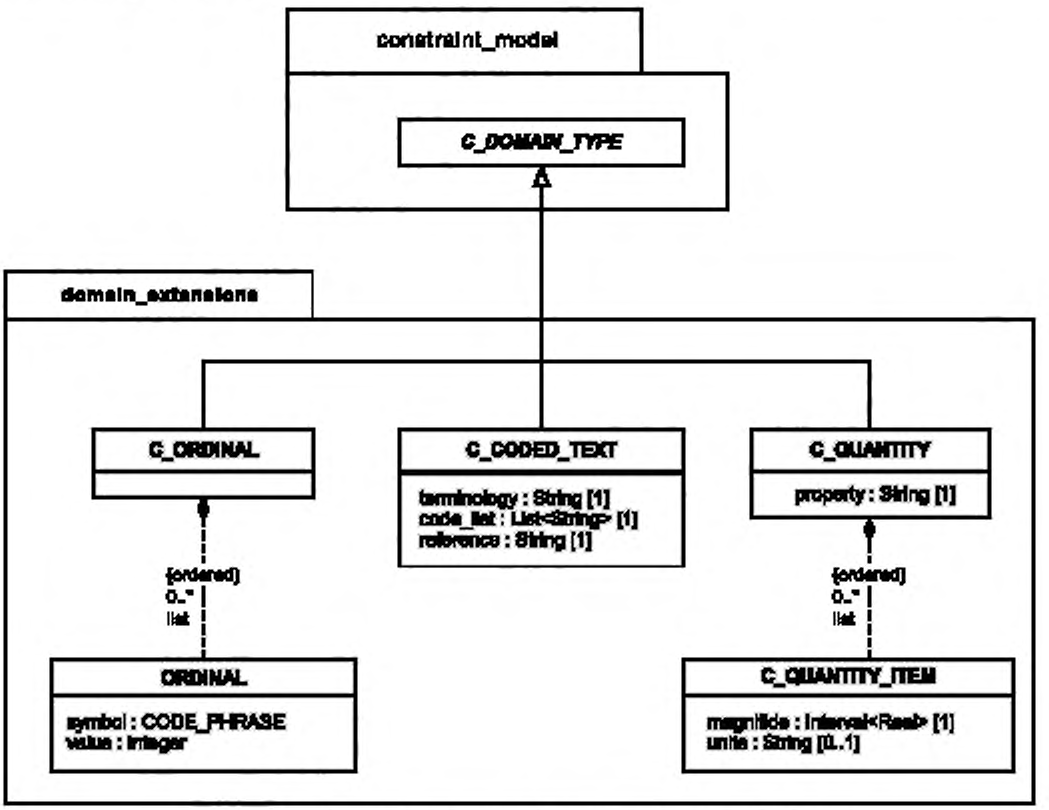

Рисунок 14 — Пример пакета, специфичного для предметной области

8 Язык определения архетипов (ADL)

8.1 Язык dADL — ADL данных

8.1.1 Обзор

8.1.1.1 Предисловие

Синтаксис языка dADL обеспечивает формальные средства описания экземпляров данных на основе базовой информационной модели, воспринимаемой как людьми, так и машинами.

Пример

```
person = List<PERSON> <
    [01234] = <
        name = <                        - - фамилия и имя лица
        forenames =                         <"Sherlock">
        family_name =                           <"Holmes">
        salutation =                            <"Mr">
        >
        address = <                             - - адрес лица
        habitation_number =                     <"221B">
```

```
            street_name =                   <Baker St>
            city =          <London>
            country =               <England>
        >
    >
    [01235] = < - - и т. д.
    >
    >
```

Примечание — В приведенном примере предполагается, что идентификаторы PERSON, name, address и т. д. взяты из информационной модели. Основной принцип конструирования языка dADL заключается в обеспечении возможности представления данных способом, равным образом пригодным как для машинной обработки, так и для восприятия человеком при минимально возможных предположениях об информационной модели, которой соответствуют данные. С этой точки зрения имена типов данных являются необязательными; часто в явном виде задаются только имена и значения атрибутов. Несколько информационных моделей могут быть совместимыми с одними и теми же данными, представленными на языке dADL. На этом языке можно выразить семантику композиций/агрегаций и ассоциаций, определенную в языке UML, как совместно используемых объектов. Значения литеральных конечных узлов могут иметь только широко распространенные типы данных, например Integer, Real, Boolean, String, Character и диапазоны даты и времени; все комплексные типы данных выражаются структурно.

8.1.1.2 Содержимое документа на языке dADL.

Документ на языке dADL может содержать один или несколько объектов из одной объектной модели.

8.1.1.3 Ключевые слова

В языке dADL нет своих собственных ключевых слов, предполагается, что все идентификаторы берутся из информационной модели.

8.1.1.4 Зарезервированные символы

В языке dADL зарезервировано несколько символов, имеющих следующие значения:

'<' — открывает объектный блок;

'>' — закрывает объектный блок;

'=' — показывает значение атрибута = объектный блок;

'(', ')' — ограничители имени типа данных или синтаксического типа подключаемого модуля;

'<#' — открывает объектный блок, выраженный в синтаксисе подключаемого модуля;

'#>' — закрывает объектный блок, выраженный в синтаксисе подключаемого модуля.

Между ограничителями <> в качестве указателей значений примитивов используются следующие символы:

'"' — символы двойных кавычек используются для ограничения строковых значений;

''' — символы одинарных кавычек используются для ограничения значений одиночных символов;

'|' — символ вертикальной черты используется для ограничения интервалов;

[ ] — квадратные скобки используются для ограничения кодированных терминов.

8.1.1.5 Комментарии

Комментарии показываются символами - -. Многострочные комментарии обозначаются символами - - в начале каждой строки комментария.

8.1.1.6 Идентификаторы информационной модели

Имя типа данных представляется идентификатором, начинающимся с прописной буквы, за которой следует любая комбинация букв, цифр и символов подчеркивания. Имя обобщенного типа данных (включая вложенные формы) может дополнительно содержать запятые и угловые скобки, но без пробелов, и должно быть синтаксически корректным по отношению к языку UML. Имя атрибута представляется любым идентификатором, начинающимся со строчной буквы, за которой следует любая комбинация букв, цифр и символов подчеркивания.

8.1.1.7 Точки с запятой

Точки с запятой факультативно используются для улучшения восприятия.

Примечание — Следующие примеры эквивалентны:

```
term = <text = < "план">;    description = <"Рекомендация врача">>
term = <text = <"план">      description = <"Рекомендация врача">>
term = <
    text = <"план">
    description = <"Рекомендация врача">
    >
```

8.1.2 Пути

Поскольку в языке dADL данные организованы иерархически и все узлы однозначно идентифицированы, то в тексте на языке dADL может быть определен уникальный путь к каждому узлу. В качестве синтаксиса путей в dADL используется стандартный синтаксис путей ADL. При использовании в данных, закодированных на языке XML, пути на языке ADL могут быть непосредственно преобразованы в выражения на языке XPath.

Примечание — Типичный путь ADL, используемый для ссылки на узел в тексте на языке dADL, выглядит следующим образом:

/ term_definitions[en]/items[at0001]/text /

8.1.3 Структура

8.1.3.1 Основная форма

8.1.3.1.1 Общая информация

Документ на языке dADL содержит сериализованные экземпляры одного или нескольких комплексных объектов. Каждый такой экземпляр является иерархией имен атрибутов и значений объектов.

Примечание — В простейшем виде текст на языке dADL состоит из повторений следующего шаблона:

имя_атрибута = <значение>

В наиболее общей форме текста на языке dADL каждое имя атрибута является именем атрибута из предполагаемой или реальной объектной или реляционной модели. Каждое значение является либо литеральным значением примитивного типа данных (см. подраздел 7.10.6), либо циклом вложений имен и значений атрибутов, завершающимся конечными узлами значений примитивного типа данных. У сестринских узлов атрибутов имена атрибутов должны быть уникальными.

Примечания

1 Следующий фрагмент демонстрирует типичную структуру:

```
attr_1 = <
    attr_2 = <
        attr_3 = <конечное_значение>
        attr_4 = <конечное_значение>
    >
    attr_5 = <
        attr_3 = <
            attr_6 = <конечное_значение>
        >
        attr_7 = <конечное_значение>
    >
>
    attr_8 = <>
```

2 В показанной выше структуре в угловые скобки заключены экземпляры некоторого типа данных. Иерархическая структура соответствует отношениям между объектами типа «является частью чего-либо»: отношениям композиции и агрегации в языке UML. Связи между экземплярами в языке dADL также представимы с помощью ссылок. Они описаны в 8.1.3.5.

8.1.3.1.2 Внешние ограничители

В тексте на языке dADL внешние ограничители <> необязательны.

8.1.3.2 Пустые секции

Пустые секции допустимы на уровне как внутренних, так и конечных узлов, позволяя автору не только выразить тот факт, что у некоторого конкретного экземпляра атрибута нет данных, но и показать, что сам атрибут ожидается существующим в базовой информационной модели. Могут использоваться вложенные пустые секции.

Пример

```
address =     <>      - - адрес лица
```

8.1.3.3 Контейнерные объекты

8.1.3.3.1 Общая информация

Экземпляры контейнеров образуются с помощью повторений блока, который начинается произвольным именем контейнерного атрибута, заключенным в квадратные скобки и квалифицируемым в каждом

случае уникальным явным значением. Квалификаторами являются произвольные уникальные ключи, которые не обязательно произведены из множества содержащихся значений. Эти ключи не обязаны быть последовательными и сами по себе не предполагаются упорядоченными. Контейнерные структуры могут появляться в любом месте структуры экземпляра.

Пример

```
school_schedule = <
    lesson_times = <08:30:00, 09:30:00, 10:30:00, ...>
    locations = <
        [1] = <under the big plane tree>
        [2] = <under the north arch>
        [3] = <in a garden>
    >
    subjects = <
        [philosophy:plato] = <      - - обратите внимание на конструкцию
                                    - - квалификатора
            name = <philosophy>
            teacher = <plato>
            topics = <meta-physics, natural science>
            weighting = <76%>
        >
        [philosophy:kant] = <
            name = <philosophy>
            teacher = <kant>
            topics = <meaning and reason, meta-physics, ethics>
            weighting = <80%>
        >
        [art] = <
            name = <art>
            teacher = <goya>
            topics = <technique, portraiture, satire>
            weighting = <78%>
        >
    >
```

8.1.3.3.2 Пути

Пути, проходящие через контейнерные объекты, формируются так же, как и пути в других структурированных данных, с добавлением ключа для обеспечения уникальности. При указании ключа в пути его значение заключается в квадратные скобки.

Пример

```
/school_schedule/locations[1]/
        - - путь к «under the big...»
/school_schedule/subjects[philosophy:kant]/
                    - - путь к «kant»
```

8.1.3.4 Добавление информации о типе данных

Необязательная информация о типе может быть указана в любом узле непосредственно перед угловой скобкой (<), открывающей любой блок, в виде идентификатора типа в стиле языка UML, в который факультативно могут быть включены идентификаторы пространств имён, разделенные точками, и параметры шаблонов. При добавлении информации о типе к данным экземпляра его имя указывается в круглых скобках после знака равенства (=).

Пример

```
destinations = <
    [seville] = (TOURIST_DESTINATION) <
    profile = (DESTINATION_PROFILE) <>
    hotels = <
        [gran sevilla] = (HISTORIC_HOTEL) <>
        [sofitel] = (LUXURY_HOTEL) <>
        [hotel real] = (PENSION) <>
    >
    attractions = <
```

```
            [la corrida] = (ATTRACTION) <>
            [Alc·zar] = (HISTORIC_SITE) <>
        >
    >
>
```

Примечание — В приведенном выше фрагменте у атрибутов "hotels" и "attractions" не указаны идентификаторы типов данных. Однако полная информация о типах может быть включена следующим образом:

```
hotels = (List<HOTEL>) <
    [gran sevilla] = (HISTORIC_HOTEL) <>
>
```

В идентификаторы типов можно включать информацию о пространстве имен, необходимую в том случае, когда одноименные типы данных определены в разных пакетах модели. Пространство имен включается в идентификатор типа с помощью указания имени пакета перед именем типа. Разделителем служит символ точки (.).

Пример

RM.EHR.CONTENT.ENTRY

и

Core.Abstractions.Relationaships.Relationship

8.1.3.5 Ассоциации и совместно используемые объекты

Ссылки на совместно используемые объекты осуществляются с использованием путей. Ссылки на объекты в других документах на языке dADL могут быть сделаны с использованием обычных идентификаторов URI, в которых фрагмент пути соответствует синтаксису пути в языке dADL.

Пример — При совместном использовании объектов "hotels" на них можно ссылаться с помощью ассоциации:

```
destinations = <
    [ "seville" ] = <
        hotels = <
            [ "gran sevilla" ] = </hotels[ "gran sevilla" ]>
            [ "sofitel" ] = </hotels[ "sofitel" ]>
            [ "hotel real" ] = </hotels[ "hotel real" ]>
        >
    >
>
bookings = <
    [ "seville:0134" ] = <
        customer_id = < "0134" >
        period = <...>
        hotel = </hotels[ "sofitel" ]>
    >
>
hotels = <
    [ "gran sevilla" ] = (HISTORIC_HOTEL) <>
    [ "sofitel" ] = (LUXURY_HOTEL) <>
    [ "hotel real" ] = (PENSION) <>
>
```

8.1.4 Конечные данные

8.1.4.1 Общая информация

В языке dADL все данные в конце концов сводятся к экземплярам примитивных типов данных String, Integer, Real, Double, Character, разным типам данных даты и времени, спискам или интервалам перечисленных типов данных и некоторым специальным типам. В языке dADL для экземпляров примитивных типов данных не используются имена типов или атрибутов, а только значения в явном виде.

8.1.4.2 Примитивные типы данных

8.1.4.2.1 Символьные данные

Символы изображаются несколькими способами. В литеральной форме символ изображается заключенным в одинарные кавычки. Символы, находящиеся вне нижнего диапазона ASCII-кодов (0-127), должны быть представлены в кодировке UTF-8.

Пример

'&ohgr;' — греческая буква «омега»

8.1.4.2.2 Строковые данные

Все строковые данные заключают в двойные кавычки. Кавычки внутри строки кодируются с использованием соглашений, описанных в стандарте ИСО/МЭК 10646.

Пример

" Это гораздо более длинная строка, которую можно назвать "предложением"."

Если надо включить в строковое значение переход на другую строку, то в него включается символ возврата каретки (return). Точное значение строкового типа данных вычисляется из символов, расположенных между символами двойных кавычек, путем удаления ведущих пробельных символов первой строки.

Пример

a ∈ A - - печатается как: a ∈ A

8.1.4.2.3 Целочисленные данные

Целочисленные данные представляются просто числами. Разделение длинных чисел запятыми и точками не допускается (см. подраздел 8.1.4.4).

Пример

25
300000
29e6

8.1.4.2.4 Действительные числа

Число считается действительным, если в его состав включена десятичная точка. Разделение длинных чисел запятыми и точками не допускается. Десятичные точки могут использоваться только для отделения дробной части числа (см. подраздел 8.1.4.4).

Пример

25.0
3.1415926
6.023e23

8.1.4.2.5 Булевские данные

Булевские значения могут быть представлены следующим образом (используемый регистр значения не имеет):

True
False

8.1.4.2.6 Даты и время

Полная дата и время

В языке dADL даты, моменты времени и длительности могут быть представлены в полной и частичной форме. Даты, время и длительности в полной форме представляются с использованием подмножества, определенного в ИСО 8601. В языке dADL способ, приведенный в использование ИСО 8601, допускается только в расширенной форме (т. е. должны использоваться символы ":" и "-"). Определенный в ИСО 8601 способ представления частичных дат, состоящих только из номера года и, в какой-то мере, из времени, включающего только число часов, не поддерживается. Допустимые частичные формы представлены ниже. Шаблоны полного представления дат и времени в языке dADL выглядят следующим образом:

yyyy-MM-dd - - дата
hh:mm[:ss[.sss][Z]] - - время
yyyy-MM-dd hh:mm:ss[.sss][Z] - - дата и время

где:

yyyy = четыре цифры года;
MM = номер месяца в году;
dd = день в месяце;
hh = час по 24-часовой шкале;
mm = минуты;
ss.sss = секунды, включая доли секунд;
Z = часовой пояс в виде знака + или –, за которым следуют четыре цифры, показывающие часовой сдвиг, например +0930, или просто литерал Z, обозначающий +0000 (Гринвичский меридиан).

Длительности выражаются в форме строки, начинающейся с символа "P", за которым следует список периодов, каждый из которых заканчивается однобуквенным обозначением: "Y" — год, "M" – месяц, "W" — неделя, "D" – день, "H" — час, "M" — минута и "S" — секунда. Символ "T" отделяет часть год-месяц-неделя-день от части час-минута-секунда, что обеспечивает различение месяцев и минут.

Пример

1919-01-23 - - дата рождения Джанго Рейнхардта
16:35.04 - - восход Венеры в Сиднее 24 июля 2003 года
2001-05-12 07:35:20+1000 - - штамп даты и времени на электронном письме - - из Австралии
P22D4H15M0S - - период длительностью 22 дня, 4 часа, 15 минут

Частичные дата и время

В языке dADL поддерживаются два способа частичного (т. е. неполного) представления даты и времени. Неполные форматы ИСО 8601 поддерживаются только в расширенной форме (т. е. с разделителями "-" и":") для всех шаблонов, которые сами по себе являются однозначными. Даты, состоящие только из года, и указания времени, состоящие только из часов, не поддерживаются. Поддерживаются следующие шаблоны ИСО 8601:

yyyy-MM - - дата без дней
hh:mm - - время без секунд
yyyy-MM-ddThh:mm - - дата и время без секунд
yyyy-MM-ddThh - - дата и время без минут и секунд

Чтобы учесть ограничения частичных шаблонов ИСО 8601 в среде контекстно-свободного синтаксического анализа, в языке dADL поддерживается вторая форма шаблонов, основанная на данных ИСО 8601. В этой форме вместо пропущенных цифр подставляются символы "?".

Следующие шаблоны соответствуют допустимым частичным датам:

yyyy-MM-?? - - дата с неизвестным днем месяца
yyyy-??-?? - - дата с неизвестными месяцем и днем

Следующие шаблоны соответствуют допустимому частичному времени:

hh:mm:?? - - время с неизвестными секундами
hh:??:?? - - время с неизвестными минутами и секундами

Следующие шаблоны соответствуют допустимой частичной дате и времени:

yyyy-MM-ddThh:mm:?? - - дата и время с неизвестными секундами
yyyy-MM-ddThh:??:?? - - дата и время с неизвестными минутами и - - секундами
yyyy-MM-ddT??:??:?? - - дата и время с неизвестным временем
yyyy-MM-??T??:??:?? - - дата и время с неизвестными днем и временем
yyyy-??-??T??:??:?? - - дата и время с неизвестными месяцем, днем и - - временем

8.1.4.3 Интервалы упорядоченных примитивных типов данных

Интервалы любого упорядоченного примитивного типа данных, т. е. Integer, Real, Date, Time, Date_Time и Duration, могут быть указаны с использованием следующего единообразного синтаксиса, где N и M обозначают экземпляры любого упорядоченного типа данных:

|N..M| двусторонний диапазон N <= x <= M;
|N<..M| двусторонний диапазон N < x <= M;
|N..<M| двусторонний диапазон N <= x < M;
|N<..<M| двусторонний диапазон N < x < M,
|<N| односторонний диапазон x < N;
|>N| односторонний диапазон x > N;
|>=N| односторонний диапазон x >= N;
|<=N| односторонний диапазон x <= N;
|N +/-M| интервал N ± M.

Допустимые значения N и M включают любые значения в диапазоне соответствующего типа данных, а также:

infinity (бесконечность)
-infinity (минус бесконечность)
* эквивалентно infinity

8.1.4.4 Другие встроенные типы данных

8.1.4.4.1 Идентификатор URI

Идентификаторы URI следуют стандартному синтаксису, определенному в документе http://www.ietf.org/rfc/rfc3986.txt. Никакие двойные или одинарные кавычки не требуются; пробелы и угловые скобки не допускаются, а если они нужны в тексте, то должны быть заключены в кавычки.

Пример

gost.e-ecolog.ru

ftp://get.this.file.com#section_5

gost.e-ecolog.ru

8.1.4.4.2 Кодированные термины

Логическая структура кодированного термина состоит из идентификатора терминологии и идентификатора кода в данной терминологии. (Связанная с кодом рубрика образует часть пакета онтологии, определенного ниже в данном разделе.) Строка в языке dADL представляется следующим образом:

[terminology_id::code]

Пример

[icd10AM::F60.1] - - код из классификатора ICD10AM
[snomed-ct::2004950] - - код из номенклатуры snomed-ct
[snomed-ct(3.1)::2004950] - - код из номенклатуры snomed-ct версии 3.1

8.1.4.5 Списки встроенных типов данных

Данные любого примитивного типа могут встречаться поодиночке или в списках, которые представляются в виде списков элементов одного типа данных, разделенных запятыми.

Пример

cyan, magenta, yellow, black - - цвета картриджей принтеров
1, 1, 2, 3, 5 - - первые 5 чисел Фибоначчи
08:02, 08:35, 09:10 - - расписание поездов

В синтаксисе никак не оговаривается, представляет ли список множество, список или какой-либо другой вид последовательности; семантика подобных структур должна определяться в базовой информационной модели.

Списки, в которых имеется только один элемент данных, представляются с использованием запятой после этого элемента, за которой следует маркер продолжения списка в виде трех точек, т. е. ...

Пример

en, ... - - языки
icd10, ... - - терминологии
[at0200], ...

В списке факультативно могут использоваться пробельные элементы.

Пример

1,1,2,3
1, 1, 2,3

8.1.5 Синтаксис языка dADL

8.1.5.1 Грамматика

Ниже определена грамматика языка dADL.

```
input:
      attr_vals
|     complex_object_block
      | error
      ;

- - - - - - - - - - - - - - - - - - - - - - - - - тело - - - - - - - - - - - - - - - - - - - - - - -
attr_vals: attr_val
      | attr_vals attr_val
      | attr_vals ';' attr_val
      ;
```

```
attr_val: attr_id SYM_EQ object_block  - - может быть один или
                                       - - несколько атрибутов
    ;
attr_id:
  V_ATTRIBUTE_IDENTIFIER
| V_ATTRIBUTE_IDENTIFIER error
object_block:
complex_object_block
|  primitive_object_block
|  plugin_object_block
plugin_object_block:
  V_PLUGIN_SYNTAX_TYPE V_PLUGIN_BLOCK
complex_object_block:
      single_attr_object_block
      | multiple_attr_object_block
      ;
multiple_attr_object_block: untyped_multiple_attr_object_block
      | TYPE_IDENTIFIER untyped_multiple_attr_object_block
      ;
untyped_multiple_attr_object_block: multiple_attr_object_block_head
                                    keyed_objects SYM_END_DBLOCK
      ;
multiple_attr_object_block_head: SYM_START_DBLOCK
      ;
keyed_objects: keyed_object
      | keyed_objects keyed_object
      ;
keyed_object: object_key SYM_EQ object_block
      ;
attr_id: V_ATTRIBUTE_IDENTIFIER
      | V_ATTRIBUTE_IDENTIFIER error
      ;
object_key: '[' simple_value ']'
      ;
single_attr_object_block: untyped_single_attr_object_block
      | TYPE_IDENTIFIER untyped_single_attr_object_block
      ;
untyped_single_attr_object_block:
      single_attr_object_complex_head SYM_END_DBLOCK
      single_attr_object_complex_head attr_vals SYM_END_DBLOCK
single_attr_object_complex_head: SYM_START_DBLOCK
      ;
primitive_object_block:
  untyped_primitive_object_block
| type_identifier untyped_primitive_object_block
untyped_primitive_object_block:
single_attr_object_primitive:
      SYM_START_DBLOCK primitive_object_value SYM_END_DBLOCK
      ;
      primitive_object_value: simple_value
      | simple_list_value
      | simple_interval_value
      | term_code
      | term_code_list_value
      | query
      ;
```

```
simple_value: string_value
      | integer_value
      | real_value
      | boolean_value
      | character_value
      | date_value
      | time_value
      | date_time_value
      | duration_value
      | uri_value
        ;
simple_list_value: string_list_value
      | integer_list_value
      | real_list_value
      | boolean_list_value
      | character_list_value
      | date_list_value
      | time_list_value
      | date_time_list_value
      | duration_list_value
        ;
simple_interval_value: integer_interval_value
     | real_interval_value
     | date_interval_value
     | time_interval_value
     | date_time_interval_value
     | duration_interval_value
        ;
type_identifier:
   V_TYPE_IDENTIFIER
|  V_GENERIC_TYPE_IDENTIFIER
- - - - - - - - - - - - - - - - - - - - - - - БАЗОВЫЕ ЗНАЧЕНИЯ ДАННЫХ - - - - - - - - - - - - - - - - - - - - - -
string_value: V_STRING
        ;
string_list_value: V_STRING ',' V_STRING
      | string_list_value ',' V_STRING
     | V_STRING ',' SYM_LIST_CONTINUE
        ;
integer_value: V_INTEGER
      | '+' V_INTEGER
      | '-' V_INTEGER
        ;
integer_list_value: integer_value ',' integer_value
     | integer_list_value ',' integer_value
     | integer_value ',' SYM_LIST_CONTINUE
        ;
integer_interval_value:
    SYM_INTERVAL_DELIM integer_value SYM_ELLIPSIS integer_value
SYM_INTERVAL_DELIM
|   SYM_INTERVAL_DELIM SYM_GT integer_value SYM_ELLIPSIS integer_value
SYM_INTERVAL_DELIM
|   SYM_INTERVAL_DELIM integer_value SYM_ELLIPSIS SYM_LT integer_value
SYM_INTERVAL_DELIM
|   SYM_INTERVAL_DELIM SYM_GT integer_value SYM_ELLIPSIS SYM_LT integer_value
SYM_INTERVAL_DELIM
```

```
| SYM_INTERVAL_DELIM SYM_LT integer_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_LE integer_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_GT integer_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_GE integer_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM integer_value SYM_INTERVAL_DELIM
    ;
real_value: V_REAL
    | '+' V_REAL
    | '–' V_REAL
    ;
real_list_value: real_value ',' real_value
    | real_list_value ',' real_value
    | real_value ',' SYM_LIST_CONTINUE
    ;
real_interval_value:
SYM_INTERVAL_DELIM real_value SYM_ELLIPSIS real_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_GT real_value SYM_ELLIPSIS real_value
SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM real_value SYM_ELLIPSIS SYM_LT real_value
SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_GT real_value SYM_ELLIPSIS SYM_LT real_value
SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_LT real_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_LE real_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_GT real_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_GE real_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM real_value SYM_INTERVAL_DELIM
    ;
boolean_value: SYM_TRUE
    | SYM_FALSE
    ;
boolean_list_value: boolean_value ',' boolean_value
    | boolean_list_value ',' boolean_value
    | boolean_value ',' SYM_LIST_CONTINUE
    ;
character_value: V_CHARACTER
    ;
character_list_value: character_value ',' character_value
    | character_list_value ',' character_value
    | character_value ',' SYM_LIST_CONTINUE
    ;
date_value: V_ISO8601_EXTENDED_DATE
date_list_value: date_value ',' date_value
    | date_list_value ',' date_value
    | date_value ',' SYM_LIST_CONTINUE
    ;
date_interval_value:
SYM_INTERVAL_DELIM date_value SYM_ELLIPSIS date_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_GT date_value SYM_ELLIPSIS date_value
SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM date_value SYM_ELLIPSIS SYM_LT date_value
SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_GT date_value SYM_ELLIPSIS SYM_LT date_value
SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_LT date_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_LE date_value SYM_INTERVAL_DELIM
```

```
| SYM_INTERVAL_DELIM SYM_GT date_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_GE date_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM date_value SYM_INTERVAL_DELIM
    ;
time_value: V_ISO8601_EXTENDED_TIME
time_list_value: time_value ',' time_value
    | time_list_value ',' time_value
    | time_value ',' SYM_LIST_CONTINUE
    ;
time_interval_value:
SYM_INTERVAL_DELIM time_value SYM_ELLIPSIS time_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_GT time_value SYM_ELLIPSIS time_value
SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM time_value SYM_ELLIPSIS SYM_LT time_value
SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_GT time_value SYM_ELLIPSIS SYM_LT time_value
SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_LT time_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_LE time_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_GT time_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_GE time_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM time_value SYM_INTERVAL_DELIM
    ;
date_time_value: V_ISO8601_EXTENDED_DATE_TIME
    ;
date_time_list_value: date_time_value ',' date_time_value
    | date_time_list_value ',' date_time_value
    | date_time_value ',' SYM_LIST_CONTINUE
    ;
date_time_interval_value:
SYM_INTERVAL_DELIM date_time_value SYM_ELLIPSIS date_time_value
SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_GT date_time_value SYM_ELLIPSIS date_time_value
SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM date_time_value SYM_ELLIPSIS SYM_LT date_time_value
SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_GT date_time_value SYM_ELLIPSIS SYM_LT
date_time_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_LT date_time_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_LE date_time_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_GT date_time_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_GE date_time_value SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM date_time_value SYM_INTERVAL_DELIM
    ;
duration_value: V_ISO8601_DURATION
    | - V_ISO8601_DURATION
duration_list_value: duration_value ',' duration_value
    | duration_list_value ',' duration_value
    | duration_value ',' SYM_LIST_CONTINUE
    ;
duration_interval_value:
SYM_INTERVAL_DELIM duration_value SYM_ELLIPSIS duration_value
SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM SYM_GT duration_value SYM_ELLIPSIS duration_value
SYM_INTERVAL_DELIM
| SYM_INTERVAL_DELIM duration_value SYM_ELLIPSIS SYM_LT duration_value
```

```
SYM_INTERVAL_DELIM
|  SYM_INTERVAL_DELIM SYM_GT duration_value SYM_ELLIPSIS SYM_LT
    ;
duration_value SYM_INTERVAL_DELIM
|  SYM_INTERVAL_DELIM SYM_LT duration_value SYM_INTERVAL_DELIM
|  SYM_INTERVAL_DELIM SYM_LE duration_value SYM_INTERVAL_DELIM
|  SYM_INTERVAL_DELIM SYM_GT duration_value SYM_INTERVAL_DELIM
|  SYM_INTERVAL_DELIM SYM_GE duration_value SYM_INTERVAL_DELIM
|  SYM_INTERVAL_DELIM duration_value SYM_INTERVAL_DELIM
term_code: V_QUALIFIED_TERM_CODE_REF
    ;
term_code_list_value: term_code ',' term_code
    | term_code_list_value ',' term_code
    | term_code ',' SYM_LIST_CONTINUE
    ;
uri_value: V_URI
```

8.1.5.2 Символы

Ниже определены символы и лексические шаблоны, используемые в приведенной выше грамматике.

```
    ------------ /* определения */ ----------------------------------------------------------
    ALPHANUM [a–zA–Z0–9]
    IDCHAR [a–zA–Z0–9_]
    NAMECHAR [a–zA–Z0–9._\–]
    NAMECHAR_SPACE [a–zA–Z0–9._\– ]
    NAMECHAR_PAREN [a–zA–Z0–9._\–()]
    UTF8CHAR (([\xC2–\xDF][\x80–\xBF])|(\xE0[\xA0–\xBF][\x80–\xBF])|([\xE1–
    \xEF][\x80–\xBF][\x80–\xBF])|(\xF0[\x90–\xBF][\x80–\xBF][\x80–\xBF])|([\xF1–
    \xF7][\x80–\xBF][\x80–\xBF][\x80–\xBF]))
    ------------ /** Разделители **/ ----------------------------------------------------------
    [ \t\r]+ - - Игнорировать разделители
    \n+ - - (увеличить счетчик строк)
    ------------ /* комментарии */ ----------------------------------------------------------
    "- -".*                                          - - Игнорировать комментарии
    "- -"».*\n[ \t\r]*
    ------------ * символы */ ----------------------------------------------------------
    -       Minus_code
    +       Plus_code
    *       Star_code
    /       Slash_code
    ^       Caret_code
    .       Dot_code
    ;       Semicolon_code
    ,        Comma_code
    :       Colon_code
    !       Exclamation_code
    (       Left_parenthesis_code
    )       Right_parenthesis_code
    $       Dollar_code
    "??"    SYM_DT_UNKNOWN
    ?       Question_mark_code
    |       SYM_INTERVAL_DELIM
    [       Left_bracket_code
    ]       Right_bracket_code
    =       SYM_EQ
    >=      SYM_GE
    <=      SYM_LE
```

```
<       SYM_LT / SYM_START_DBLOCK
>       SYM_GT / SYM_END_DBLOCK
..      SYM_ELLIPSIS
...     SYM_LIST_CONTINUE
------------ /* ключевые слова */ ----------
[Tt][Rr][Uu][Ee]                                    SYM_TRUE
[Ff][Aa][Ll][Ss][Ee]                                SYM_FALSE
[Ii][Nn][Ff][Ii][Nn][Ii][Tt][Yy]                    SYM_INFINITY
[Qq][Uu][Ee][Rr][Yy]                                SYM_QUERY_FUNC
------------ /* идентификаторы URI */ ----------
[a–z]+:\/\/[ ^<>|\\{}^~"\[\]]*                                  V_URI
------------ /* ссылка на код термина в форме [ICD10AM(1998)::F23 */ ----------
\[{NAMECHAR_PAREN}+::{NAMECHAR_SPACE}+\] V_QUALIFIED_TERM_CODE_REF
\[{ALPHANUM}{NAMECHAR}*\]          V_LOCAL_TERM_CODE_REF
------------ /* определение местного кода */ ----------
a[ct][0-9.]+                                                    V_LOCAL_CODE
------------ /* V_ISO8601_EXTENDED_DATE_TIME YYYY-MM-DDThh:mm:ss[,sss][Z|+/–
nnnn] */ ---
[0–9]{4}–[0–1][0–9]–[0–3][0–9]T[0–2][0-9]:[0–6][0–9]:[0–6][0–9](,[0–
9]+)?(Z|[+–][0–9]{4})? |
[0–9]{4}–[0–1][0–9]–[0–3][0–9]T[0–2][0–9]:[0–6][0–9](Z|[+–][0–9]{4})? |
[0–9]{4}–[0–1][0–9]–[0–3][0–9]T[0–2][0–9](Z|[+–][0–9]{4})?
------------ /* V_ISO8601_EXTENDED_TIME hh:mm:ss[,sss][Z|+/–nnnn] */ ----------
[0–2][0–9]:[0–6][0–9]:[0–6][0–9](,[0–9]+)?(Z|[+–][0–9]{4})? |
[0–2][0–9]:[0–6][0–9](Z|[+–][0–9]{4})?
------------ /* V_ISO8601_EXTENDED_DATE YYYY–MM–DD */ ----------
[0–9]{4}–[0–1][0–9]–[0–3][0–9] |
[0–9]{4}–[0–1][0–9]
------------ /* V_ISO8601_DURATION PnYnMnWnDTnnHnnMnnS */ ----------
P([0–9]+[yY])?([0–9]+[mM])?([0–9]+[wW])?([0–9]+[dD])?T([0–9]+[hH])?([0–
9]+[mM])?([0–9]+[sS])? |
P([0-9]+[yY])?([0–9]+[mM])?([0–9]+[wW])?([0–9]+[dD])?
------------ /* V_TYPE_IDENTIFIER */ ----------
[A–Z]{IDCHAR}*
------------ /* V_GENERIC_TYPE_IDENTIFIER */ ----------
[A–Z]{IDCHAR}*<[a–zA–Z0-9,_<>]+>
------------ /* V_ATTRIBUTE_IDENTIFIER */ ----------
[a–z]{IDCHAR}*
------------ /* Блоки CADL */ ----------
\{[^{}]*
<IN_CADL_BLOCK>\{[^{}]*          - - получена открывающая фигурная скобка
<IN_CADL_BLOCK>[^{}]*\}          - - получена закрывающая фигурная скобка
------------ /* числа */ ----------
[0–9]+\.[0–9]+                                  V_INTEGER
[0–9]+\.[0–9]+[eE][+–]?[0–9]+                   V_REAL
------------ /* Строки */ ----------
\"[^\\\n"]*\"                                   V_STRING
---- строки, содержащие кавычки, специальные символы и т. д.
\"[^\\\n"]*                     - - начало строки
<IN_STR>\\\\                    - - соответствует обратной косой черте
<IN_STR>\\\"                    - - соответствует двойной кавычке
{UTF8CHAR}+                     - - соответствует символам UTF8
<IN_STR>[^\\\n"]+               - - соответствует любым другим символам
<IN_STR>\\\n[ \t\r]*            - - соответствует символу перехода к новой
                                - - строке (LF)
```

```
<IN_STR>[^\n"]*\"          - - соответствует окончательному концу
                           - - строкового значения
<IN_STR>.|\n|              - - ошибка
<IN_STR><<EOF>>            - - незавершенная строка
- - - - - - - - - - - - - /* V_CHARACTER */ - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
\'[^\\\n']\' — обычный символ из диапазона 0–127
\'\\n\ - - \n
\'\\r\ - - \r
\'\\t\ - - \t
\'\\'\ - - \'
\'\\\ - - \\
\'{UTF8CHAR}\' — символ UTF8
\'.{1,2}|
\'\\[0–9]+(\./)? — недопустимый символ –> ERR_CHARACTER
```

8.2 Язык cADL — ADL ограничений

8.2.1 Обзор (справочно)

Синтаксис языка cADL позволяет описывать ограничения данных, определенных с помощью объектно-ориентированных информационных моделей, в архетипах или в других формализмах определения знаний. Данный язык наиболее подходит для определения специфичных допустимых конструкций данных, экземпляры которых соответствуют очень общим объектным моделям. Язык cADL используется как на этапе разработки (разработчиками и/или инструментальными средствами), так и во время исполнения вычислительными системами, которые проверяют правильность данных, сравнивая их с соответствующими разделами архетипа, описывающими ограничения на языке cADL. Общий вид ограничений на языке cADL иллюстрируется следующим примером:

```
PERSON[at0000] matches {        - - ограничение экземпляра PERSON
    name matches {              - - ограничение PERSON.name
        TEXT matches {/.+/}     - - любая непустая строка
    }
    addresses cardinality matches {0..*} matches        { - - ограничение
    ADDRESS matches {                                     - - PERSON.addresses
            - - и т.д. - -
        }
    }
}
```

Некоторые текстовые ключевые слова в данном примере могут быть эффективно представлены обычными символами математической логики. В следующем примере ключевое слово matches заменено соответствующим символом:

```
PERSON[at0000] ∈ {              - - ограничение экземпляра PERSON
    name ∈ {                    - - ограничение PERSON.name
      TEXT ∈ {/..*/}            - - любая непустая строка
    }
    addresses cardinality ∈ {0..*} ∈  { - - ограничение
        ADDRESS ∈ {               - - PERSON.addresses
            - - и т. д. - -
        }
    }
}
```

Вся совокупность эквивалентов представлена ниже. Необработанная форма языка cADL хранится в текстовом виде, чтобы исключить возможные трудности с представлением специальных символов и избежать трудностей обработки текста на языке cADL в текстовых редакторах, не обеспечивающих восприятие таких символов, а также облегчить чтение текста на английском языке. Однако символьная форма представления может быть более употребительной благодаря использованию инструментальных средств, форматированию текстов в формате HTML или в других форматах представления документов. Кроме того, символьная форма может быть более удобна пользователям, не владеющим английским языком, а также имеющим хорошую математическую подготовку. Язык cADL поддерживает оба варианта представления — использование символов или текста зависит только от предпочтений пользователя.

Литеральные конечные значения (например регулярное выражение /..*/ в приведенном выше примере) всегда являются ограничениями множества стандартных примитивных типов данных. Другие более сложные типы ограничений описаны ниже.

8.2.2 Основные сведения

8.2.2.1 Ключевые слова

В языке cADL используются следующие ключевые слова:

- matches, ~matches, is_in, ~is_in;
- occurrences, existence, cardinality;
- ordered, unordered, unique;
- infinity;
- use_node, allow_archetype*;
- include, exclude.

Символьные эквиваленты некоторых из приведенных выше ключевых слов показаны в следующей таблице:

Текстовое представление	Символьное представление	Значение
matches, is_in	∈	Членство во множестве, элемент p принадлежит множеству P
not, ~	~	Отрицание, не p

Оператор matches или is_in является ключевым оператором языка cADL; он соответствует в математике членству во множестве. Его указание между именем и блоком, ограниченным фигурными скобками, означает следующее: внутри фигурных скобок задано множество допустимых значений элемента с указанным именем (объекта или частей объекта — атрибутов). То, что находится внутри любой пары фигурных скобок, соответствующих друг другу, может рассматриваться как спецификация множества значений. Так как блоки бывают вложенными, то данный подход к указанию значений можно интерпретировать в терминах вложенных множеств или в терминах пространства значений объектов, имеющих типы данных, принадлежащие определенной совокупности.

Примечания

1 В приведенном ниже примере оператор matches связывает имя объекта с линейным пространством значений (например, со списком), образованном всеми словами, оканчивающимися на «ion»:

```
aaa matches {/.*ion[^\s\n\t]/}  - - множество английских слов, оканчивающихся
                                - - на «ion»
```

2 В следующем примере имя типа данных XXX связано с комплексным многомерным пространством значений:

```
XXX matches {
    aaa matches {                       - -
        YYY matches {0..3}              - -
    }                                   - - пространство значений
    bbb matches {                       - - и экземпляр XXX
        ZZZ matches {>1992-12-01}       - -
    }                                   - -
}
```

3 Иногда требуется, чтобы оператор matches использовался в отрицании; обычно это имеет место в конечном блоке. Любой из следующих фрагментов может быть использован для ограничения пространства значений элемента XXX любым числом, кроме 5:

```
XXX ~matches {5}
XXX ~is_in {5}
XXX ∉ {5}
```

8.2.2.2 Комментарии

Комментарии в языке cADL обозначаются двойным символом дефиса '- -'. Многострочный комментарий формируется из нескольких строк, каждая из которых начинается с ведущих символов '- -'.

---

* Существовало еще ключевое слово use_archetype, которое сейчас не используется.

8.2.2.3 Идентификаторы в информационной модели

Именем типа данных является любой идентификатор, начинающийся с прописной буквы, за которой следует любая комбинация из букв, цифр и символов подчеркивания. Имя обобщенного типа данных (включая вложенные формы) может дополнительно содержать запятые и угловые скобки, но не пробелы, и должно быть синтаксически корректным по отношению к языку UML. Именем атрибута является любой идентификатор, начинающийся со строчной буквы, за которой следует любая комбинация из букв, цифр и символов подчеркивания.

8.2.2.4 Идентификаторы узлов

В языке cADL элемент, заключенный в квадратные скобки, например [xxxx], используется для идентификации узлов объектов, т. е. узлов, описывающих ограничения экземпляров некоторого типа данных. Узлы объектов всегда начинаются с имени типа данных. Внутри квадратных скобок может появиться любая строка в зависимости от ее использования.

8.2.2.5 Естественный язык

Язык cADL не зависит от естественного языка. Единственным потенциальным исключением является случай, когда ограничения включают литеральные значения на некотором языке, что может быть обойдено при использовании раздельных языковых и терминологических определений. Однако в целях улучшения восприятия читателем в настоящий стандарт включены комментарии на русском языке.

8.2.3 Структура

8.2.3.1 Общая информация

Ограничения в языке cADL представлены в стиле структуры блоков. Общая структура представляет собой вложенные ограничения типов данных, за которыми следуют ограничения свойств (относящихся к данному типу), типы данных (относящиеся к атрибутам) и т. д. Термин «блок объекта» или «узел объекта» относится к любому блоку, начинающемуся именем типа данных (записанному полностью прописными буквами), а термин «блок атрибута» или «узел атрибута» относится к любому блоку, начинающемуся идентификатором атрибута (записанному полностью строчными буквами).

Примечания

1 Типичный блок выглядит следующим образом (повторяющийся шаблон /.+/ является регулярным выражением, представляющим непустую строку):

```
PERSON [at0001] ∈ {
    name ∈ {
        PERSON_NAME [at0002] ∈ {
            forenames cardinality ∈ {1..*} ∈ {/.+/}
            family_name ∈ {/.+/}
            title ∈ {Dr, Miss, Mrs, Mr, ...}
        }
    }
    addresses cardinality ∈ {1..*} ∈ {
        LOCATION_ADDRESS [at0003] ∈ {
            street_number existence ∈ {0..1} ∈ {/.+/}
            street_name ∈ {/.+/}
            locality ∈ {/.+/}
            post_code ∈ {/.+/}
            state ∈ {/.+/}
            country ∈ {/.+/}
        }
    }
}
```

2 В приведенном выше фрагменте любой идентификатор (выделенный полужирным шрифтом), за которым следуют оператор включения ∈ (эквивалентный ключевому слову matches или is_in) и открывающая фигурная скобка, является началом блока, простирающегося до соответствующей закрывающей фигурной скобки (для лучшего восприятия текста закрывающая скобка располагается с тем же смещением относительно начала строки, что и начало блока). Приведенный пример описывает ограничение экземпляра типа данных PERSON; ограничением является все то, что записано внутри блока PERSON. Два блока, расположенные на следующем уровне, определяют ограничения свойства экземпляра типа PERSON, в данном случае этими свойствами являются names и addresses.

8.2.3.2 Комплексные объекты

Ограничения, сформулированные на языке cADL, не могут быть сильнее тех, что описаны в информационной модели, ограниченной архетипом. Более того, текст на языке cADL включает ограничения только тех частей модели, которые разумно ограничивать.

Примечание — В следующем примере показано, как можно сформулировать ограничение свойства value класса ELEMENT, который имеет тип QUANTITY с диапазоном, подходящим для описания кровяного давления:

```
ELEMENT [at0010] matches {   - - диастолическое кровяное давление
        value matches {
            QUANTITY matches {
                magnitude matches {0..1000}
                property matches {"pressure"}
                units matches {"mm[Hg]"}
            }
        }
}
```

8.2.3.3 Ограничения атрибутов

8.2.3.3.1 Общая информация

В любой базовой информационной модели атрибуты могут иметь либо одно, либо несколько значений, то есть принадлежать к обобщенному контейнерному типу, например List<Contact>.

8.2.3.3.2 Существование

Ограничение существования может быть указано непосредственно после любого идентификатора атрибута; оно показывает, является ли объект, к которому относится значение атрибута, обязательным или необязательным для данных. Смысл ограничения существования в том, является ли соответствующий объект или атрибут обязательным или необязательным для данных экземпляра. Такая же логика применяется и для определения, имеет ли атрибут единичную или множественную кратность, т. е. является ли он контейнером или нет. Для контейнерных атрибутов ограничение существования указывает, является ли весь контейнер (обычно это список или множество) обязательным или нет; последующее ограничение кратности (описанное ниже) показывает, сколько элементов допускается в данном контейнере. Существование изображается с использованием того же языка ограничений, что и в остальном определении архетипа. Ограничения существования могут принимать значения {0}, {0..0}, {0..1}, {1} или {1..1}. Значением ограничения существования по умолчанию, если нет специальных указаний, является {1..1}.

Примечание — Ограничения существования выражаются в языке cADL следующим образом:

```
QUANTITY matches {
        units existence matches {0..1} matches {mm[Hg]}
}
```

8.2.3.4 Атрибуты с единственным значением

Повторяющиеся блоки ограничений объектов одного класса (или его подтипов) могут иметь в языке cADL два возможных значения в зависимости от того, указана ли кратность в блоке описания атрибута. Два или более блоков объектов, начинающихся именем типа данных, появляющиеся после атрибута, не являющегося контейнером (т. е. не имеющего ограничения кратности), воспринимаются как альтернативные ограничения, только одно из которых должно применяться к данным.

Пример

```
ELEMENT[at0004] matches { - - ограничение скорости
        value matches {
            QUANTITY matches {
                magnitude matches {|0..55|}
                property matches {"velocity"}
                units matches {"mph"} - - миль/ч
            }
            QUANTITY matches {
                magnitude matches {|0..100|}
                property matches {"velocity"}
                units matches {"km/h"} - - км/ч
            }
        }
}
```

Примечание — В данном примере кратность атрибута value (по умолчанию) равна 1..1, а наличие двух ограничений типа данных QUANTITY не является обязательным, и в данных времени исполнения может появиться только один экземпляр типа QUANTITY, соответствующий любому из этих ограничений.

8.2.3.5 Контейнерные атрибуты

8.2.3.5.1 Кратность

Контейнерные атрибуты представляются на языке cADL с ограничением кратности. Кратность указывает ограничение числа элементов контейнерных типов данных, например списков или множеств.

Пример

```
HISTORY [at0001] occurrences ∈ {1} ∈ {
    periodic ∈ {False}
    events cardinality ∈ {*} ∈ {
        EVENT [at0002] occurrences ∈ {0..1} ∈ { }
                - - оценка 1 мин
        EVENT [at0003] occurrences ∈ {0..1} ∈ { }
                - - оценка 2 мин
        EVENT [at0004] occurrences ∈ {0..1} ∈ { }
                - - оценка 3 мин
    }
}
```

Ограничение кратности может быть задано после имени атрибута (или после ограничения его существования, если таковое имеется). Оно указывает, что данный атрибут имеет контейнерный тип данных, какое число элементов он может иметь, и, возможно, имеет ли он семантику списка, множества или пакета с помощью ключевых слов ordered, unordered, unique и non-unique. Целочисленный диапазон используется для указания допустимого числа элементов контейнера; одиночный символ * означает диапазон 0..*, т. е. от нуля до любого числа.

Пример

```
events cardinality ∈ {*; ordered} ∈ {          - - логический список
events cardinality ∈ {*; unordered; unique} ∈ {  - - логическое множество
events cardinality ∈ {*; unordered} ∈ {         - - логический пакет
```

Ограничения кратности и существования могут использоваться совместно, указывая разные комбинации ограничений свойства контейнерного типа данных.

Пример

```
events existence ∈ {0..1} cardinality ∈ {0..*} ∈ {- - и т. д. - -}
```

8.2.3.5.2 Число вхождений

Ограничение числа вхождений (occurrences) может использоваться только для узлов объектов, описанных на языке cADL (но не для узлов атрибутов). Оно указывает, сколько раз экземпляр данного класса, к которому применяется данное ограничение, может встречаться в данных времени исполнения. Это ограничение имеет смысл только для объектов, являющихся потомками контейнерного атрибута, так как по определению число вхождений объекта, являющегося значением однозначного атрибута, может быть только 0..1 или 1..1, что уже определено ограничением существования данного атрибута. Однако допускается указывать это ограничение и в других случаях. Значением ограничения occurrences по умолчанию, если иное не указано, является {1..1}.

Примеры

1 Ниже показаны три ограничения объекта EVENT — первое (оценка 1 мин) показано как обязательное, а два других — как необязательные:

```
events cardinality ∈ {*} ∈ {
    EVENT [at0002] occurrences ∈ {1..1} ∈ { }      - - оценка 1 мин
    EVENT [at0003] occurrences ∈ {0..1} ∈ { }      - - оценка 2 мин
    EVENT [at0004] occurrences ∈ {0..1} ∈ { }      - - оценка 3 мин
}
```

2 Ниже представлено ограничение экземпляров объекта GROUP, согласно которому экземпляры, описывающие племена, клубы и семьи, должны иметь только один заголовок, но могут иметь много элементов:

```
GROUP [at0103] ∈ {
        kind ∉ {/tribe|family|club/}
        members cardinality ∈ {*} ∈ {
            PERSON [at0104] occurrences ∈ {1} matches {
                title ∉ {head}
                - - и т. д. - -
            }
            PERSON [at0105] occurrences ∉ {0..*} matches {
                title ∈ {member}
                - - и т. д. - -
            }
        }
}
```

8.2.3.6 Ограничение «любое значение»

Ограничение «любое значение» обозначается одиночным символом (*). С его помощью можно в явном виде указать, что некоторое свойство может иметь любое значение.

Пример — В приведенном ниже фрагменте ограничение «любое значение» атрибута name означает, что любое значение, разрешенное базовой информационной моделью, также разрешено архетипом; однако при этом также обеспечивается возможность указать ограничение существования, которое может быть уже того, что описано в информационной модели:

```
PERSON [at0001] matches {
        name existence matches {0..1} matches {*}
        - - и т. д. - -
}
```

Ограничение «любое значение» может также использоваться для указания того, что свойство value объекта ELEMENT должно иметь конкретный тип данных, но при этом может иметь любое значение, допустимое для этого типа.

Пример

```
ELEMENT[at0004] matches { - - ограничение скорости
        value matches {
            QUANTITY matches {*}
        }
}
```

8.2.3.7 Идентификация узлов объектов и путей

Идентификатор узла требуется для любого узла объекта, на который предполагается сделать ссылку где-либо в тексте на языке cADL или в системе времени исполнения, и который в противном случае был бы неоднозначным (то есть у него имеются сестринские узлы).

Пример

```
members cardinality ∈ {*} ∈ {
    PERSON [at0104] ∉ {
        title ∉ {"head"}
    }
    PERSON [at0105] matches {
        title ∈ {"member"}
    }
}
```

Всем узлам в тексте на языке cADL, соответствующим узлам в данных, на которые могут быть ссылки из других мест архетипа, или которые могут быть использованы в запросах времени исполнения, должны быть присвоены идентификаторы узла. Идентификатор узла может также использоваться для придания узлу смыслового значения, если в качестве его имени выбирается некоторое описание.

Пути в языке cADL предназначены для ссылок на узлы текста на языке cADL. Для задания пути используется стандартный синтаксис пути в языке ADL, подробно описанный в разделе 8.4. Пути в языке ADL имеют такую же структуру чередующихся пар объект/атрибут, какие применяют в общей иерархической структуре текста на языке cADL, реализующей шаблон ТИП/атрибут/ТИП/атрибут/... Пути в языке cADL всегда указывают на узлы объектов и могут быть созданы только для узлов, имеющих идентификатор узла, или узлов, являющихся единственным дочерним узлом атрибута единичной кратности. Путь всегда должен заканчиваться разделителем «косая черта» (/).

Что необычно для синтаксиса пути, может потребоваться указание идентификатора конечного объекта, даже если свойство соответствует единственному отношению (как это имеет место для свойства "имени" объекта), поскольку язык cADL позволяет указывать несколько альтернативных ограничений объектов (каждый из которых идентифицируется уникальным идентификатором узла) для узла отношения, имеющего единичную кратность.

Пример

```
HISTORY occurrences ∈ {1} ∈ {
        periodic ∈ {FALSE}
        events cardinality ∈ {*} ∈ {
            EVENT [at0002] occurrences ∈ {1..1} ∈ { } - - оценка 1 мин
            EVENT [at0003] occurrences ∈ {0..1} ∈ { } - - оценка 2 мин
            EVENT [at0004] occurrences ∈ {0..1} ∈ { } - - оценка 3 мин
        }
}
```

К узлам этого примера могут быть построены следующие пути:

```
/                       - - объект HISTORY
/periodic               - - атрибут HISTORY.periodic
/events[at0002]         - - объект события 1 мин
/events[at0003]         - - объект события 2 мин
/events[at0004]         - - объект события 3 мин
```

В конце пути можно добавлять ссылки на атрибуты, если это разрешено в базовой информационной модели.

Пример

```
/events/count           - - атрибут count свойства элементов
```

Физические пути могут преобразовываться в логические пути с использованием описательных смысловых идентификаторов узлов, если таковые определены.

Пример — Следующие два пути эквивалентны:

```
/events[at004]              - - объект события 3 мин
/events[3 minute event]     - - объект события 3 мин
```

Для ссылки на узел текста на языке cADL из какого-либо места в архетипе требуется, чтобы идентификатор исходного архетипа был добавлен в префикс пути.

Пример

```
[openehr-ehr-entry.apgar-result.v1]/events[at0002]
```

8.2.3.8 Внутренние ссылки в архетипе

Часто требуется указать ограничение, которое по существу повторяет предшествующее комплексное ограничение, но в другом блоке. Это можно сделать с помощью указания в архетипе внутренней ссылки согласно следующему правилу.

Внутренняя ссылка в архетипе, обеспечивающая повторение комплексного ограничения, ранее определенного в том же архетипе, задается с помощью ключевого слова use_node в строке следующего вида:

use_node ТИП путь_к_объекту

Пример

```
PERSON [at0001] ∈ {
        identities ∈ {
          - - etc - -
        }
        contacts cardinality ∈ {0..*} ∈ {
          CONTACT [at0002] ∈ {                          - - домашний адрес
                purpose ∈ {- - и т. д. - -}
                addresses ∈ {- - и т. д. - -}
          }
          CONTACT [at0003] ∈ {                          - - почтовый адрес
                purpose ∈ {- - и т. д. - -}
                addresses ∈ {- - и т. д. - -}
          }
          CONTACT [at0004] ∈ {                          - - домашняя контактная информация
                purpose ∈ {- - и т. д. - -}
                addresses cardinality ∈ {0..*} ∈ {
                    ADDRESS [at0005] ∈ {                - - телефон
                          type ∈ {- - и т. д. - -}
                          details ∈ {- - и т. д. - -}
                    ADDRESS [at0006] ∈ {                - - факс
                          type ∈ {- - и т.д. - -}
                          details ∈ {- - и т. д. - -}
                    }
                    ADDRESS [at0007] ∈ {                - - электронная почта
                          type ∈ {- - и т. д. - -}
                          details ∈ {- - и т. д. - -}
                     }
          }
          }
          CONTACT [at0008] ∈ {                  - - рабочая контактная
                                                - - информация
                          purpose ∈ {- - и т. д. - -}
                          addresses cardinality ∈ {0..*} ∈ {
use_node ADDRESS /[at0001]/contacts[at0004]/addresses[at0005]/ - - телефон
use_node ADDRESS /[at0001]/contacts[at0004]/addresses[at0006]/ - - факс
use_node ADDRESS /[at0001]/contacts[at0004]/addresses[at0007]/ - - эл.почта
                    }
          }
        }
```

8.2.3.9 Слоты архетипов

Слот архетипа задается с помощью ключевого слова allow_archetype и определяется с помощью двух списков утверждений, начинающихся ключевыми словами include и exclude, соответственно. Это позволяет не определять заново требуемые ограничения, а использовать вместо них ранее определенные архетипы. Для слота архетипа задаются два списка утверждений, указывающих, какие архетипы допускаются в этом слоте и/или какие не могут быть в него вставлены.

Слот может быть широким, то есть в него допускается вставлять много других архетипов, или узким, если в него можно вставить только несколько архетипов или всего лишь один архетип. Место в архетипе, в котором определен слот, является связующей точкой.

Пример — Ниже показано, как в разделе объективных данных архетипа заголовков проблем пациента определяются два слота, указывающие, какие разделы ENTRY и SECTION архетипов допускаются или исключаются из свойства элементов items:

```
SECTION [at2000] occurrences ∈ {0..1} ∈ {
                        - - объективные данные
    items ∈ {
        allow_archetype ENTRY occurrences ∈ {0..1} ∈ {
          include
              concept_short_name ∈ {/.+/}
        }
```

```
            allow_archetype SECTION occurrences ∈ {0..*} ∈ {
                include
                    id ∈ {/.*\.iso-ehr\.section\..*\..*/}
                exclude
                    id ∈ {/.*\.iso-ehr\.section\.patient_details\..*/}
            }
        }
}
```

В ограничении слота архетипа можно указать, что допустимый архетип должен содержать определенное ключевое слово или определенный путь.

8.2.3.10 Смешанные структуры

Выше были описаны три типа структур, описывающих ограничения комплексных объектов:

- структуры комплексных объектов — любой узел, начинающийся именем типа, за которым следуют фигурные скобки, внутри которых содержатся ограничения атрибутов, инварианты и т. д.;

- внутренние ссылки — ссылка на любой узел, начинающаяся ключевым словом use_node, за которым следует имя типа; такие узлы соответствуют ограничению комплексного объекта, определенного в этом же архетипе ранее;

- слоты архетипов — любой узел, начинающийся ключевым словом allow_archetype, за которым следует имя типа; такие узлы задают ограничение комплексного объекта, определенное в каком-либо другом архетипе.

В любом узле все три типа структур могут сосуществовать.

Пример

```
SECTION [at2000] ∈ {
        items cardinality ∈ {0..*; ordered} ∈ {
            ENTRY [at2001] ∈ {- - и т. д. - -}
            allow_archetype ENTRY ∈ {- - и т. д. - -}
            use_node ENTRY [at0001]/some_path[at0004]/
            ENTRY [at2002] ∈ {- - и т. д. - -}
            use_node ENTRY /[at1002]/some_path[at1012]/
            use_node ENTRY /[at1005]/some_path[at1052]/
            ENTRY[at2003] ∈ {- - и т. д. - -}
        }
}
```

8.2.4 Ограничения примитивных типов данных

8.2.4.1 Общая информация

В языке cADL ограничения атрибутов, имеющих примитивные типы данных, могут факультативно формулироваться без имени типа. Один уровень фигурных скобок может быть опущен.

Пример

```
some_attr matches {some_pattern}
    вместо:
some_attr matches {
        PRIMITIVE_TYPE matches {
            some_pattern
        }
}
```

8.2.4.2 Ограничения строк

8.2.4.2.1 Общая информация

Строки могут быть ограничены двумя способами: с использованием фиксированной строки и с использованием регулярного выражения. Все ограничения строк чувствительны к регистру ввода.

8.2.4.2.2 Список строк

Атрибут со строковым значением может быть ограничен списком строк (с использованием синтаксиса языка dADL для списков строк), включая простую одиночную строку.

Пример

```
species matches {"platypus"}
species matches {"platypus", "kangaroo"}
species matches {"platypus", "kangaroo", "wombat"}
```

Примечание — Первый пример ограничивает значение времени исполнения атрибута species некоторого объекта значением "platypus"; второй пример ограничивает его значением "platypus" или "kangaroo" и т. д. Почти во всех случаях данный вид строкового ограничения не должен применяться, так как он делает тело архетипа зависимым от языка, за исключением собственных имен, которые обычно стандартизованы в международном масштабе.

8.2.4.2.3 Регулярное выражение

Вторым способом ограничения строк являются регулярные выражения. Синтаксис регулярных выражений допустим подмножеством синтаксиса, используемого в языке Perl. В языке cADL приняты три варианта синтаксиса:

```
string_attr matches {/regular expression/}
string_attr matches {=~ /regular expression/}
string_attr matches {!~ /regular expression/}
```

Первые два варианта идентичны. Они указывают на то, что значение атрибута должно соответствовать данному регулярному выражению. Третий вариант указывает на то, что значение не должно соответствовать данному выражению.

Если в шаблоне требуется ограничивающий символ, то перед ним должен быть поставлен символ «обратная косая черта» (\). Можно также применять другие ограничители, с которыми шаблоны будут более понятными.

Примечание — Типичным примером являются регулярные выражения, включающие единицы измерения; ниже приведены два эквивалентных шаблона:

```
units matches {/km\ /h|mi\ /h/}
units matches {^km/h|mi /h^}
```

В языке cADL поддерживаются следующие шаблоны регулярных выражений.

Атомарные элементы

. точка соответствует любому одиночному символу, например, / ... / соответствует любым трем символам, расположенным между пробелами;

[xyz] соответствует любому символу из набора xyz (с учетом регистра), например, /[0–9]/ соответствует любой строке, содержащей одну десятичную цифру;

[a–m] соответствует любому из символов сплошного диапазона от a до m (с учетом регистра), например, /[0–9]/ соответствует любой строке, состоящей из одиночного символа, представляющего десятичную цифру. /[S–Z]/ соответствует любому одиночному символу из диапазона S – Z;

[^a–m] соответствует любому символу, к роме тех, что попадают в сплошной диапазон от a до m, например, /[^0–9]/ соответствует любой строке, состоящей из одиночного символа, которым не является десятичная цифра.

Группировка

(pattern) для группировки элементов используются круглые скобки; любой шаблон, расположенный внутри круглых скобок, рассматривается как атомарный элемент для операторов вхождения, например, /([0–9][0–9])/ соответствует любому двузначному числу.

Вхождения

* соответствует ни одному или нескольким вхождениям предшествующего атомарного элемента, например, /.*/ соответствует любой непустой строке, /[a–z]*/ соответствует любой непустой строке английских букв на нижнем регистре;

+ соответствует одному или нескольким вхождениям предшествующего атомарного элемента, например, /a.+/ соответствует любой строке, начинающейся с символа a, за которым следует, по крайней мере, еще один символ;

? соответствует ни одному или одному вхождению предшествующего атомарного элемента, например, /ab?/ соответствует строкам a и ab;

{m,n} соответствует от m до n вхождениям предшествующего атомарного элемента, например, /ab{1,3}/ соответствует строкам ab, abb и abbb, /[a–z]{1, 3}/ соответствует всем буквенным строкам на нижнем регистре длиной от одного до трех символов;

{m,} соответствует, по крайней мере, m вхождениям предшествующего атомарного элемента;

{,n} соответствует не более чем n вхождениям предшествующего атомарного элемента;

{m} соответствует точно m вхождениям предшествующего атомарного элемента.

Специальные классы символов

\d соответствует символу десятичной цифры;
\D соответствует нецифровому символу;
\s соответствует пробельному символу;
\S соответствует не пробельному символу.

Альтернативы

шаблон1|шаблон2 соответствует либо шаблон1, либо шаблон2, например, /лежащий|сидящий|стоящий/ соответствует любому из слов "лежащий", "сидящий" или "стоящий".

8.2.4.3 Ограничения целых чисел

Целые числа могут быть ограничены одним целочисленным значением, целочисленным интервалом или списком целых чисел.

Пример

```
length matches {1000}          - - указывает интервал 1000 (= фиксированному
                               - - значению)
length matches {|950..1050|}   - - допускает 950 — 1050
length matches {|0..1000|}     - - допускает 0 — 1000
length matches {|0..<1000|}    - - допускает 0 <= x < 1000
length matches {|0<..<1000|}   - - допускает 0 < x < 1000
length matches {|<=10|}        - - допускает не больше чем 10
length matches {|>=10|}        - - допускает 10 или больше
length matches {|100+/–5|}     - - допускает 100 ± 5, т. е. отрезок 95 — 105
rate matches {|0..infinity|}   - - допускает 0 — бесконечность, т. е. то же
                               - - самое, что и >= 0
```

8.2.4.4 Ограничения действительных чисел

Ограничения действительных чисел подчиняются тем же синтаксическим правилам, что и целочисленные ограничения, за исключением того, что все действительные числа изображаются с десятичной точкой и, по крайней мере, с одной последующей цифрой, которой может быть 0.

Пример

```
magnitude matches {5.5}          - - фиксированное значение
magnitude matches {|5.5|}        - - указывает интервал (=фиксированному
                                 - - значению)
magnitude matches {|5.5..6.0|}   - - интервал
magnitude matches {5.5, 6.0, 6.5} - - список
magnitude matches {|0.0..<1000.0|} - - допускает 0.0 >= x <1000.0
magnitude matches {|>10.0|}      - - допускает больше чем 10.0
magnitude matches {|<=10.0|}     - - допускает не больше чем 10.0
magnitude matches {|>=10.0|}     - - допускает 10.0 или больше
magnitude matches {|80.0+/–12.0|} - - допускает 80.0 ± 12.0
```

8.2.4.5 Ограничения булевских значений

Булевские значения времени исполнения могут быть ограничены значениями True (истина), False (ложь) или принимать любое из этих значений:

some_flag matches {True}
some_flag matches {False}
some_flag matches {True, False}

8.2.4.6 Ограничения символов

8.2.4.6.1 Общая информация

Символы могут быть ограничены двумя способами: списком символов и регулярным выражением.

8.2.4.6.2 Списки символов

Значение символа может быть ограничено списком фиксированных значений. Каждый символ заключается в одинарные кавычки.

Пример

```
color_name matches {'r'}
color_name matches {'r', 'g', 'b'}
```

8.2.4.6.3 Регулярные выражения

Значения символов могут быть ограничены с помощью элементов односимвольного регулярного выражения, также заключенного в одинарные кавычки.

Пример

color_name matches {'[rgbcmyk]'}
color_name matches {'[^\s\t\n]'}

В символьных выражениях допускаются только следующие элементы синтаксиса регулярных выражений:

- любой элемент из списка атомарных элементов из 8.2.4.2.3;
- любой элемент из списка элементов специальных классов из 8.2.4.2.3;
- символ, обозначающий любой символ;
- альтернативное выражение, части которого являются любыми типами данных элементов, например, a|b|[m–z].

8.2.4.7 Ограничения дат, времени и длительности

8.2.4.7.1 Общая информация

Даты, время, даты и время, а также длительности могут быть ограничены тремя способами: списками значений, интервалами и шаблонами.

8.2.4.7.2 Дата, время, дата и время

Шаблоны

Даты, время, даты и время (т. е. отметки времени) могут быть ограничены с использованием шаблонов, основанных на синтаксисе даты/времени из ИСО 8601, которыей указывают, какие части даты или времени должны быть заданы. Шаблон ограничения формируется из абстрактного шаблона yyyy-mm-ddThh:mm:ss (который сам получен с помощью преобразования каждого поля даты/времени из ИСО 8601 в букву, представляющую тип поля) с подстановкой в соответствующие места символов "?" (означающих необязательность) или "X" (означающих запрет). Упрощенная грамматика шаблона может быть представлена следующим образом (в расширенной форме Бэкуса-Наура ENBF все показанные символы являются литералами):

ограничение_даты:	yyyy – mm|??|XX – dd|??|XX
ограничение_времени:	hh : mm|??|XX : ss|??|XX
ограничение_времени_в_дате:	T hh|??|XX : mm|??|XX : ss|??|XX
ограничение_даты_и_времени:	ограничение_даты ограничение_времени_в_дате

Все выражения, построенные на основе данной грамматики, должны также удовлетворять следующим правилам:

если какое-то поле содержит символы "??", то поля справа от него могут содержать только символы "??" или "XX";

если какое-то поле содержит символы "XX", то поля справа от него могут содержать только символы "XX".

Для реализации как упрощенной грамматики, так и правил проверки может быть определена более полная грамматика.

В приведенной ниже таблице показаны варианты допустимых шаблонов и типы данных, подразумеваемые каждым шаблоном:

Подразумеваемый тип данных	Шаблон	Пояснение
Дата	yyyy–mm–dd	Должна быть указана полная дата
Дата, частичная дата	yyyy–mm–??	Необязательный день, например день месяца забыт
Дата, частичная дата	yyyy –??–??	Необязательные месяц и день, т. е. допустима любая дата, например медицинские вопросники для оценки умственного развития включают хорошо известные исторические даты
Частичная дата	yyyy–??–XX	Необязательный месяц, нет дня
Время	Thh:mm:ss	Должно быть указано полное время

Окончание таблицы

Подразумеваемый тип данных	Шаблон	Пояснение
Частичное время	Thh:mm:XX	Нет секунд, например, назначенное время встречи
Частичное время	Thh:??:XX	Необязательные минуты, нет секунд, например обычное время на часах
Время, частичное время	Thh:??:??	Необязательные минуты и секунды, т. е. любое допустимое время
Дата и время	yyyy–mm–ddThh:mm:ss	Должны быть указаны полные дата и время
Дата и время, частичные дата и время	yyyy –mm–ddThh:mm:??	Необязательные секунды, например назначенные дата и время
Частичные дата и время	yyyy–mm–ddThh:mm:XX	Нет секунд, например назначенные дата и время
Частичные дата и время	yyyy–mm–ddThh:??:XX	Нет секунд, необязательные минуты, например вспоминаемые пациентом дата и время
Дата и время, частичные дата и время, частичная дата и частичное время	yyyy–??–??T??:??:??	Минимум ограничений на задание даты и времени

Интервалы

Даты, время, даты и время могут быть также ограничены интервалами. Каждая дата, время и т. д. в интервале может быть литеральной датой, временем и т. д., значением или значением, основанным на шаблоне. В последнем случае ограничивающие значения задаются с помощью шаблонов из приведенной выше таблицы, но с числами в позициях, где нет символов "X" или "?".

Пример

|1995-??-XX| - - любая частичная дата в 1995 г.
|09:30:00| - - точно 9:30
|< 09:30:00| - - любое время до 9:30
|<= 09:30:00| - - любое время до 9:30 или 9:30
|> 09:30:00| - - любое время после 9:30
|>= 09:30:00| - - 9:30 или любое время после 9:30
|2004-05-20..2004-06-02| - - диапазон дат
|2004-05-20T00:00:00..2005-05-19T23:59:59| - - диапазон дат и времени

8.2.4.7.3 Ограничения длительности

Шаблоны

Шаблоны, основанные на ИСО 8601, могут использоваться для ограничения длительностей так же, как и для дат и времени. Обобщенная форма шаблона выглядит следующим образом (в форме EBNF, все символы являются литералами):

P[Y|y][M|m][W|w][D|d][T[H|h][M|m][S|s]]

Следует отметить, что обозначение "W", используемое вместе с другими обозначениями, является отклонением от опубликованного в ИСО 8601. Обозначение "W" (неделя) может использоваться вместе с другими обозначениями, так как широко распространено определение длительности режима или лечения в виде комбинации недель и дней. Данный шаблон указывает, какие могут быть заполнены «слоты» из строки ИСО, задающей длительность. Если в данном шаблоне указано несколько символов, то это означает «или», т. е. любой один или несколько слотов могут быть представлены в данных.

Пример

Pd - - длительность, содержащая только дни, например P5d
Pm - - длительность, содержащая только месяцы, например P5m
PTm - - длительность, содержащая только минуты, например PT5m
Pwd - - длительность, содержащая только недели и/или дни, например P4w
PThm - - длительность, содержащая только часы и/или минуты, например PT2h30m

Списки и интервалы

Длительности могут также ограничиваться абсолютными значениями, удовлетворяющими данным ИСО 8601, или диапазонами.

Пример

PT1m - - 1 мин
P1dT8h - - 1день, 8 ч
|PT0m..PT1m30s| - - приемлемый временной сдвиг от первой оценки по шкале Апгар

8.2.4.8 Ограничения списков примитивных типов данных

Во многих случаях ограничиваемым типом данных в информационной модели атрибута является список или множество примитивных типов данных. Такое ограничение должно быть следующим образом указано в языке cADL с использованием ключевого слова cardinality (как для комплексных типов данных):

some_attr cardinality matches {0..*} matches {some_pattern}

Шаблон, которому должны соответствовать окончательные результаты, будет содержать список или множество ограничений значений, а не единственное ограничение значения. При этом для однозначных атрибутов может использоваться любое описанное выше ограничение, которое соответствует типу данных рассматриваемого атрибута. Однако, как и в случае комплексных объектов, каждый элемент списка должен принимать одно из значений, получаемых из формулировки ограничения.

Примечание — В следующем примере каждый элемент списка, соответствующего значению атрибута speed_limits (имеющего тип данных List<Integer>), ограничен одним из чисел 50, 60, 70 и т. д.:

speed_limits cardinality matches {0..*; ordered} matches {50, 60, 70, 80, 100, 130}.

8.2.4.9 Подразумеваемые значения

Чтобы пользователи или системы знали, какое значение подразумевается в том случае, если необязательные элементы не включены в данные, такие значения можно явным образом задать в описании архетипа. Подразумеваемые значения можно задавать только для примитивных типов данных; при записи они начинаются символом точки с запятой, за которым следует значение того же типа данных, что и в предшествующей части ограничения.

Пример

length matches {|0..1000|; 200} - - допускает 0 — 1000, подразумевается 200
some_flag matches {True, False; True } - - допускает T или F, подразумевается T
some_date matches {yyyy-mm-dd hh:mm:XX; 1800-01-01 00:00:00}

Если подразумеваемые значения не заданы, то на основании анализа архетипа получатель его содержания не может сделать никакого достоверного предположения о значениях отсутствующих необязательных частей.

8.2.5 Синтаксис языка cADL

8.2.5.1 Грамматика

Ниже определена грамматика языка cADL.

```
input:
  c_complex_object
| error
c_complex_object:
  c_complex_object_head SYM_MATCHES SYM_START_CBLOCK c_complex_object_body
c_invariants SYM_END_CBLOCK
c_complex_object_head:
  c_complex_object_id c_occurrences
c_complex_object_id:
  TYPE_IDENTIFIER
| TYPE_IDENTIFIER V_LOCAL_TERM_CODE_REF
c_complex_object_body:
  c_any
| c_attributes
```

```
c_object:
   c_complex_object
        | archetype_internal_ref
        | archetype_slot
        | constraint_ref
        | c_coded_term
        | c_ordinal
        | c_primitive_object
        | V_C_DOMAIN_TYPE
        | ERR_C_DOMAIN_TYPE
        | error
archetype_internal_ref:
   SYM_USE_NODE TYPE_IDENTIFIER_C_OCCURRENCES object_path
|  SYM_USE_NODE TYPE_IDENTIFIER error
archetype_slot:
   c_archetype_slot_head SYM_MATCHES SYM_START_CBLOCK c_includes
   c_excludes
SYM_END_CBLOCK
c_archetype_slot_head:
   c_archetype_slot_id c_occurrences
c_archetype_slot_id:
   SYM_ALLOW_ARCHETYPE TYPE_IDENTIFIER
|  SYM_ALLOW_ARCHETYPE TYPE_IDENTIFIER V_LOCAL_TERM_CODE_REF
|  SYM_ALLOW_ARCHETYPE error
c_primitive_object:
   c_primitive
c_primitive:
   c_integer
| c_real
| c_date
| c_time
| c_date_time
| c_duration
| c_string
| c_boolean
| error
c_any:
   *
c_attributes:
   c_attribute
|  c_attributes c_attribute
c_attribute:
   c_attr_head SYM_MATCHES SYM_START_CBLOCK c_attr_values SYM_END_CBLOCK
c_attr_head:
   V_ATTRIBUTE_IDENTIFIER c_existence
|  V_ATTRIBUTE_IDENTIFIER c_existence c_cardinality
c_attr_values:
   c_object
|  c_attr_values c_object
|  c_any
|  error
c_includes:
   -/-
|  SYM_INCLUDE invariants
```

```
c_excludes:
   -/-
| SYM_EXCLUDE invariants
c_existence:
   -/-
| SYM_EXISTENCE SYM_MATCHES SYM_START_CBLOCK existence_spec SYM_END_CBLOCK
existence_spec:
   V_INTEGER
|  V_INTEGER SYM_ELLIPSIS V_INTEGER
c_cardinality:
   SYM_CARDINALITY SYM_MATCHES SYM_START_CBLOCK cardinality_spec
SYM_END_CBLOCK
cardinality_spec:
   occurrence_spec
|  occurrence_spec ; SYM_ORDERED
|  occurrence_spec ; SYM_UNORDERED
|  occurrence_spec ; SYM_UNIQUE
|  occurrence_spec ; SYM_ORDERED ; SYM_UNIQUE
|  occurrence_spec ; SYM_UNORDERED ; SYM_UNIQUE
|  occurrence_spec ; SYM_UNIQUE ; SYM_ORDERED
|  occurrence_spec ; SYM_UNIQUE ; SYM_UNORDERED
cardinality_limit_value:
   integer_value
|  *
c_occurrences:
   -/-
| SYM_OCCURRENCES SYM_MATCHES SYM_START_CBLOCK occurrence_spec
SYM_END_CBLOCK
| SYM_OCCURRENCES error
occurrence_spec:
   cardinality_limit_value
| V_INTEGER SYM_ELLIPSIS cardinality_limit_value
c_integer_spec:
   integer_value
|  integer_list_value
|  integer_interval_value
|   occurrence_spec
c_integer:
   c_integer_spec
|  c_integer_spec ; integer_value
|  c_integer_spec ; error
c_real_spec:
   real_value
|  real_list_value
|  real_interval_value
c_real:
   c_real_spec
|  c_real_spec ; real_value
|  c_real_spec ; error
c_date_constraint:
   V_ISO8601_DATE_CONSTRAINT_PATTERN
|  date_value
|  date_interval_value
```

```
c_date:
   c_date_constraint
|  c_date_constraint ; date_value
|  c_date_constraint ; error
c_time_constraint:
   V_ISO8601_TIME_CONSTRAINT_PATTERN
|  time_value
|  time_interval_value
c_time:
   c_time_constraint
|  c_time_constraint ; time_value
|  c_time_constraint ; error
c_date_time_constraint:
   V_ISO8601_DATE_TIME_CONSTRAINT_PATTERN
|  date_time_value
|  date_time_interval_value
c_date_time:
   c_date_time_constraint
|  c_date_time_constraint ; date_time_value
|  c_date_time_constraint ; error
c_duration_constraint:
   V_ISO8601_DURATION_CONSTRAINT_PATTERN
|  duration_value
|  duration_interval_value
c_duration:
   c_duration_constraint
|  c_duration_constraint ; duration_value
|  c_duration_constraint ; error
c_string_spec:
   V_STRING
|  string_list_value
|  string_list_value , SYM_LIST_CONTINUE
|  V_REGEXP
c_string:
   c_string_spec
|  c_string_spec ; string_value
|  c_string_spec ; error
c_boolean_spec:
   SYM_TRUE
|  SYM_FALSE
|  SYM_TRUE , SYM_FALSE
|  SYM_FALSE , SYM_TRUE
c_boolean:
   c_boolean_spec
|  c_boolean_spec ; boolean_value
|  c_boolean_spec ; error
constraint_ref:
   V_LOCAL_TERM_CODE_REF
any_identifier:
   TYPE_IDENTIFIER
|  V_ATTRIBUTE_IDENTIFIER
```

8.2.5.2 Символы

Ниже определена лексическая спецификация грамматики языка cADL.

```
- - - - - - - - - - - - /* определения */ - - - - - - - - - - -
```

```
IDCHAR [a–zA–Z0–9_]
NAMECHAR [a–zA–Z0–9._\–]
NAMECHAR_SPACE [a–zA–Z0–9._\– ]
NAMECHAR_PAREN [a–zA–Z0–9._\–()]
UTF8CHAR (([\xC2–\xDF][\x80–\xBF])|(\xE0[\xA0–\xBF][\x80–\xBF])|([\xE1–\xEF][\x80–\xBF][\x80–
\xBF])|(\xF0[\x90–\xBF][\x80–\xBF][\x80–\xBF])|([\xF1–\xF7][\x80–\xBF][\x80–\xBF][\x80–\xBF]))
------------ /* комментарии */ ------------------------------------------------------------
"- -".*                                    - - Игнорировать комментарии
"- -".*\n[ \t\r]*
------------ /* символы */ ---------------------------------------------------------------
"–"          Minus_code
"+"          Plus_code
"*"          Star_code
"/"          Slash_code
"^"          Caret_code
"="          Equal_code
"."          Dot_code
";"          Semicolon_code
"."          Comma_code
":"          Colon_code
"!"          Exclamation_code
"("          Left_parenthesis_code
")"          Right_parenthesis_code
"$"          Dollar_code
"??"         SYM_DT_UNKNOWN
"?"          Question_mark_code
"|"          SYM_INTERVAL_DELIM
"["          Left_bracket_code
"]"          Right_bracket_code
"{"          SYM_START_CBLOCK
"}"          SYM_END_CBLOCK
".."         SYM_ELLIPSIS
"..."        SYM_LIST_CONTINUE
------------ /* общие ключевые слова */ ---------------------------------------------------
[Mm][Aa][Tt][Cc][Hh][Ee][Ss]              SYM_MATCHES
[Ii][Ss]_[Ii][Nn]                         SYM_MATCHES
------------ /* ключевые слова утверждений */ ---------------------------------------------
[Tt][Hh][Ee][Nn]                                          SYM_THEN
[Ee][Ll][Ss][Ee]                                          SYM_ELSE
[Aa][Nn][Dd]                                              SYM_AND
[Oo][Rr]                                                  SYM_OR
[Xx][Oo][Rr]                                              SYM_XOR
[Nn][Oo][Tt]                                              SYM_NOT
[Ii][Mm][Pp][Ll][Ii][Ee][Ss]                              SYM_IMPLIES
[Tt][Rr][Uu][Ee]                                          SYM_TRUE
[Ff][Aa][Ll][Ss][Ee]                                      SYM_FALSE
[Ff][Oo][Rr][ _ ] [Aa][Ll][Ll]                            SYM_FORALL
[Ee][Xx][Ii][Ss][Tt][Ss]                                  SYM_EXISTS
[Ee][Xx][Ii][Ss][Tt][Ee][Nn][Cc][Ee]                      SYM_EXISTENCE
[Oo][Cc][Cc][Uu][Rr][Rr][Ee][Nn][Cc][Ee][Ss]              SYM_OCCURRENCES
[Cc][Aa][Rr][Dd][Ii][Nn][Aa][Ll][Ii][Tt][Yy]              SYM_CARDINALITY
[Oo][Rr][Dd][Ee][Rr][Ee][Dd]                              SYM_ORDERED
[Uu][Nn][Oo][Rr][Dd][Ee][Rr][Ee][Dd]                      SYM_UNORDERED
[Uu][Nn][Ii][Qq][Uu][Ee]                                  SYM_UNIQUE
[Ii][Nn][Ff][Ii][Nn][Ii][Tt][Yy]                          SYM_INFINITY
```

```
[Uu][Ss][Ee][ _ ][Nn][Oo][Dd][Ee]                                   SYM_USE_NODE
[Uu][Ss][Ee][ _ ][Aa][Rr][Cc][Hh][Ee][Tt][Yy][Pp][Ee]               SYM_ALLOW_ARCHETYPE
[Aa][Ll][Ll][Oo][Ww][ _ ][Aa][Rr][Cc][Hh][Ee][Tt][Yy][Pp][Ee]       SYM_ALLOW_ARCHETYPE
[Ii][Nn][Cc][Ll][Uu][Dd][Ee]                                        SYM_INCLUDE
[Ee][Xx][Cc][Ll][Uu][Dd][Ee]                                        SYM_EXCLUDE
- - - - - - - - - - - - /* V_URI */ - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
[a-z]+:\ /\ /[^<>|\\{}^~'"\[\] ]*{
- - - - - - - - - - - - /* V_QUALIFIED_TERM_CODE_REF */ - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - любой подходящий код, например [local::at0001], [local::ac0001], [loinc::700–0]
\[{NAMECHAR_PAREN}+::{NAMECHAR}+\]
\[{NAMECHAR_PAREN}+::{NAMECHAR_SPACE}+\]                            - - ошибка
- - - - - - - - - - - - /* V_TERM_CODE_CONSTRAINT of form */ - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - [terminology_id::code, - - комментарий
- - code, - - комментарий
- -  code] - - комментарий
- -
- - Форма с подразумеваемым значением
- - [terminology_id::code, - - комментарий
- -  code; - - комментарий
- -  code] — необязательное подразумеваемое значение
- -
\[[a-zA-Z0-9()._\–]+::[ \t\n]* - - начало IN_TERM_CONSTRAINT
<IN_TERM_CONSTRAINT> {
            [ \t]*[a-zA-Z0-9._\–]+[ \t]*:[ \t\n]*
- - соответствует второй снизу строке, оканчивающейся ':' (подразумеваемое значение)
            [ \t]*[a–zA–Z0–9._\–]+[ \t]*,[ \t\n]*
- - соответствует любой строке, оканчивающейся запятой ','
            \–\–[^\n]*\n            - - игнорировать комментарии
            [ \t]*[a–zA–Z0–9._\–]*[ \t\n]*\]
- - соответствует последней строке, оканчивающейся скобкой ']'
- - - - - - - - - - - - /* V_LOCAL_TERM_CODE_REF */ - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - любой не квалифицированный код, например [at0001], [ac0001], [700-0] - -
\[{ALPHANUM}{NAMECHAR}*\]
- - - - - - - - - - - - /* V_LOCAL_CODE */ - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
a[ct][0–9.]+
- - - - - - - - - - - - /* V_QUALIFIED_TERM_CODE_REF */ - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - любой квалифицированный код, например [local::at0001], [local::ac0001],
- - [loinc::700–0]
\[{NAMECHAR_PAREN}+::{NAMECHAR}+\]
\[{NAMECHAR_PAREN}+::{NAMECHAR_SPACE}+\]                            - - ошибка
\[[a–zA–Z - - - - - - - - - /* V_ISO8601_EXTENDED_DATE_TIME */ - - -
- - YYYY–MM–DDThh:mm:ss[,sss][Z|+/–nnnn]
- -
[0–9][a–zA–Z]{4} – [0–1][0–9._\–]*\]
- - - - - - - - /* V_LOCAL_CODE */ - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
a[ct]–[0-3][0–9.]+
- - - - - - - /* V_QUALIFIED_TERM_CODE_REF */ - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - любой квалифицированный код, например [local::at0001], [local::ac0001],
- - [loinc::700–0]
\[[a–zA–Z]T[0–2][0–9().._\–]+::[a–zA–Z]:[0–6][0–9._\–]+\] ]:[0–6][0–
9](,[0–9]+)?(Z|[+–][0–9]{4})? |
[0–9]{4} – [0–1][0–9]–[0–3][0–9]T[0–2][0–9]:[0–6][0–9](Z|[+–][0–9]{4})? |
[0–9]{4} – [0–1][0–9]–[0–3][0–9]T[0–2][0–9](Z|[+–][0–9]{4})?
- - - - - - - - /* V_ISO8601_EXTENDED_TIME */ - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

```
- - hh:mm:ss[.sss][Z|+/–nnnn]
- -
[0–2][0–9]:[0–6][0–9]:[0–6][0–9](.[0–9]+)?(Z|[+–][0–9]{4})? |
[0–2][0–9]:[0–6][0–9](Z|[+–][0–9]{4})?
- - - - - - - - - - /* V_ISO8601_DATE YYYY–MM–DD */- - - - - - - - - -
```

```
[0–9]+\.[0–9]+[eE][+–]?[0–9]+
- - - - - - - - - /* V_STRING */ - - - - - - - -
```

8.3.3.3 Операторы равенства

Поддерживаются следующие операторы равенства:

- равенство: = ;
- неравенство: <> .

Семантика этих операторов заключается в сравнении значений.

8.3.3.4 Операторы отношения

Поддерживаются следующие операторы отношения:

- меньше: < ;
- меньше или равно: <= ;
- больше: > ;
- больше или равно: >= .

Семантика этих операторов заключается в сравнении значений. Их область применения ограничена значениями сопоставимых типов данных.

8.3.3.5 Булевские операторы

Поддерживаются следующие булевские операторы:

- не: **not**;
- и: **and**;
- исключительное или: **xor**;
- имеет следствием: **implies**;
- принадлежность к множеству: **matches, is_in**.

Булевские операторы также имеют символьные эквиваленты, показанные выше.

8.3.3.6 Кванторы

Поддерживаются два стандартных кванторных оператора.

- квантор существования: **exists**;
- квантор всеобщности: **for_all**.

Данные операторы также имеют обычные символьные эквиваленты, показанные выше.

8.3.4 Операнды

Операнды формулировок утверждений могут быть следующих типов:

- именованная константа — любая константа любого примитивного типа данных, записанная в соответствии с синтаксисом языка dADL для значений;
- ссылка на переменную — любое имя, начинающееся с символа $, например $body_weight;
- ссылка на свойство — путь к свойству, т. е. любой путь, заканчивающийся на .имя_свойства;
- ссылка на объект — путь к узлу объекта, т. е. любой путь, заканчивающийся идентификатором узла.

Если утверждение используется в определении слота архетипа, то его операнды относятся к архетипу, заполняющему слот, а не к архетипу, содержащему данный слот.

8.3.5 Переменные

8.3.5.1 Предопределенные переменные

В формулировках утверждений на языке ADL можно указывать ряд предопределенных переменных, включая следующие:

- $current_date: Date; возвращает текущую дату при каждом использовании архетипа;
- $current_time: Time; возвращает текущее время при каждом использовании архетипа;
- $current_datetime: Date_Time; возвращает текущие дату и время при каждом использовании архетипа.

8.3.5.2 Переменные, определенные в архетипе

Переменные могут быть также определены внутри архетипа как часть формулировок утверждений в секции invariant. Принят следующий синтаксис определения переменной:

let $var_name = reference

При этом reference может иметь любой тип из перечисленных выше типов операндов. Определения переменных могут встречаться в любом месте блока invariant, но для улучшения читаемости текста их, как правило, следует приводить вначале.

Примечание — Следующий пример иллюстрирует использование переменных в блоке invariant:

```
invariant
    let $sys_bp =
        /data[at9001]/events[at9002]/data[at1000]/items[at1100]
    let $dia_bp =
        /data[at9001]/events[at9002]/data[at1000]/items[at1200]
    $sys_bp >= $dia_bp
```

8.3.6 Грамматика

```
assertions:
    assertion
|   assertions assertion
assertion:
    any_identifier : boolean_expression
|   boolean_expression
|   any_identifier : error
boolean_expression:
    boolean_leaf
|   boolean_node
boolean_node:
    SYM_EXISTS absolute_path
|   SYM_EXISTS error
|   relative_path SYM_MATCHES SYM_START_CBLOCK c_primitive SYM_END_CBLOCK
|   SYM_NOT boolean_leaf
|   arithmetic_expression = arithmetic_expression
|   arithmetic_expression SYM_NE arithmetic_expression
|   arithmetic_expression SYM_LT arithmetic_expression
|   arithmetic_expression SYM_GT arithmetic_expression
|   arithmetic_expression SYM_LE arithmetic_expression
|   arithmetic_expression SYM_GE arithmetic_expression
|   boolean_expression SYM_AND boolean_expression
|   boolean_expression SYM_OR boolean_expression
|   boolean_expression SYM_XOR boolean_expression
|   boolean_expression SYM_IMPLIES boolean_expression
boolean_leaf:
    ( boolean_expression )
|   SYM_TRUE
|   SYM_FALSE
arithmetic_expression:
    arithmetic_leaf
|   arithmetic_node
arithmetic_node:
    arithmetic_expression + arithmetic_leaf
|   arithmetic_expression – arithmetic_leaf
|   arithmetic_expression * arithmetic_leaf
|   arithmetic_expression / arithmetic_leaf
|   arithmetic_expression ^ arithmetic_leaf
arithmetic_leaf:
    ( arithmetic_expression )
|   integer_value
|   real_value
    | absolute_path
```

8.4 Пути в языке ADL

8.4.1 Обзор

Понятие путей является интегральной частью языка ADL, а общий синтаксис путей используется для ссылок на узлы в секциях архетипа как на языке dADL, так и на языке cADL. Один и тот же синтаксис используется в обоих языках, так как и dADL, и cADL имеют одинаковую структуру чередующихся объектов и атрибутов. Однако интерпретация выражений для путей в dADL и cADL немного различается; разница поясняется в 8.1.2 и 8.2.3. В данном подразделе представлены только общие синтаксис и семантика.

Обобщенная форма синтаксиса пути выглядит следующим образом:

/][ид_объекта/]{имя_атрибута[ид_объекта]/}*

Пути в языке ADL формируются с помощью чередования сегментов, состоящих из имени атрибута и необязательного предиката идентификатора узла объекта, разделенных символами «косой черты» (/). Идентификаторы узлов заключены в квадратные скобки ([ ]). Путь завершается или косой чертой, и тогда он идентифицирует узел объекта, или именем атрибута, и тогда он идентифицирует узел атрибута.

Пути могут быть абсолютными или задаваться относительно узла, в котором они упоминаются. Абсолютные пути всегда начинаются косой чертой.

8.4.2 Синтаксис пути

8.4.2.1 Грамматика

input:
movable_path
| absolute_path
| relative_path
| error

movable_path:
SYM_MOVABLE_LEADER relative_path

absolute_path:
/ relative_path
| absolute_path / relative_path

relative_path:
path_segment
| relative_path / path_segment

path_segment:
V_ATTRIBUTE_IDENTIFIER V_LOCAL_TERM_CODE_REF
| V_ATTRIBUTE_IDENTIFIER

8.4.2.2 Символы

"."	Dot_code
"/"	Slash_code
"["	Left_bracket_code
"]"	Right_bracket_code
"//"	SYM_MOVABLE_LEADER
\[[a–zA–Z0–9][a–zA–Z0–9._\–]*\]	V_LOCAL_TERM_CODE_REF
[A–Z][a–zA–Z0–_]*	V_TYPE_IDENTIFIER
[a–z][a–zA–Z0–9_]*[ ]*\(\)	V_FEATURE_CALL_IDENTIFIER
[a–z][a–zA–Z0–9_]*	V_ATTRIBUTE_IDENTIFIER

8.5 ADL – язык определения архетипов

8.5.1 Общая информация

В данном подразделе описан общий вид архетипов на языке ADL с добавлением небольших подробностей к уже приведенным описаниям языков dADL и cADL. Подробно представлена важная тема взаимосвязи секции definition, закодированной на языке cADL, с секцией ontology, закодированной на языке dADL.

Архетип на языке ADL имеет следующую структуру:

archetype
archetype_id
[specialize
parent_archetype_id]
concept
coded_concept_name
language
dADL language description section
description
dADL metadata section
definition
cADL structural section

invariant
    assertions
ontology
    dADL definitions section
[revision_history
    dADL section]

8.5.2 Основы

8.5.2.1 Ключевые слова

В языке ADL имеется небольшое число представленных ниже ключевых слов, зарезервированных для использования в объявлениях архетипов:

- archetype, specialize/specialize, concept,
- description, definition, ontology.

Все эти ключевые слова могут использоваться как идентификаторы в секциях definition и ontology.

8.5.2.2 Идентификация узла

В секции definition архетипа на языке ADL используется особая схема кодирования идентификаторов узлов, а также обозначений ограничений текстовых элементов (зависящих от языка). Коды могут быть местными по отношению к архетипу или заимствованными из внешнего лексикона. Это означает, что описание архетипа будет одинаковым на всех языках и доступным на любом языке, на который коды были переведены. Все коды терминов заключены в квадратные скобки ([ ]). К кодам, используемым в качестве идентификаторов узлов и определенным в том же архетипе, добавляется префикс at, и по соглашению они имеют четыре цифры, например [at0010]. В архетипах на языке ADL допускаются коды любой длины. Специализации локально кодированных понятий имеют единый корень, за которым следуют расширения, отделяемые точкой, например [at0010.2]. С терминологической точки зрения такие коды не имеют встроенной семантики – структурирование с помощью точек используется как оптимизация идентификации узлов.

8.5.2.3 Местные коды ограничений

Местные коды используются также для обозначения ограничений текстовых элементов в теле архетипа. Хотя эти ограничения могли бы быть включены в основное тело архетипа, но они чувствительны к языку и/или терминологии и поэтому определяются в секции ontology и обозначаются кодами с префиксом ac, например [ac0009]. Как и для кодов с префиксом at, в настоящем стандарте принято соглашение представлять коды ограничений состоящими из четырех цифр, хотя и допускается любое число цифр в коде. Использование данных кодов описано в подпункте 8.5.6.4.

8.5.3 Секции заголовка

8.5.3.1 Секция archetype (архетип)

Данная секция объявляет архетип и должна содержать идентификатор. В глобальном пространстве архетипы обозначаются фасетными идентификаторами.

Примечание — Типичная секция archetype выглядит следующим образом:

```
archetype (adl_version=1.4)
    mayo.openehr-ehr-entry.haematology.v1
```

8.5.3.2 Признак контроля версий

Флаг, показывающий, применяется ли к архетипу контроль версий или нет, может быть указан после версии, например:

```
archetype (adl_version=1.4; controlled)
  mayo.openehr-ehr-entry.haematology.v1
```

Данный флаг может иметь только два значения «controlled (контролируемый)» и «uncontrolled (неконтролируемый)» и предназначен для помощи программному обеспечению. В архетипы с флагом «контролируемый» должна быть включена секция контроля версий, а в архетипах с флагом «неконтролируемый» или вообще без флага история версий может быть опущена. Это дает возможность неофициально редактировать архетипы на ранней стадии разработки, не создавая длинные истории версий, имеющие небольшое значение или вообще никакого.

8.5.3.3 Секция specialize (специализация)

Эта необязательная секция показывает, что данный архетип является специализацией некоторого другого архетипа, идентификатор которого должен быть указан. Допускается только один специализируемый архетип-родитель, то есть архетип не может иметь множественное наследование от других архетипов.

Примечание — Ниже представлен пример объявления специализации, в котором идентификатор нового архетипа получается путем добавления нового суффикса к имени предметной области родителя:

```
archetype (adl_version=1.4)
    mayo.openehr-ehr-entry.haematology-cbc.v1
specialise
    mayo.openehr-ehr-entry.haematology.v1
```

В языке ADL допускается использовать как английское, так и американское правописание слова specialize/specialise.

8.5.3.4 Секция concept (понятие)

Все архетипы представляют понятия реального мира, например «пациент», «кровяное давление» или «дородовое наблюдение». Понятие всегда кодируется, чтобы его можно было воспроизвести на любом языке, на который архетип был переведен.

Примечание — В следующем примере определением термина [at0010] является описание, соответствующее секции haematology-cbc приведенного выше архетипа:

```
concept
    [at0010]                - - результат клинического исследования крови
```

8.5.3.5 Секция language (язык) и перевод на другие языки

Секция language содержит данные, описывающие исходный язык, на котором архетип был создан (существенные для оценки качества переводов), и общий список языков, доступных в данном архетипе. Может существовать только один исходный язык original_language. Список переводов translations должен обновляться всякий раз, когда добавляется новый перевод архетипа.

Пример

```
language
    original_language = <"en">
    translations = <
        ["de"] = <
            provenance = <"gost.e-ecolog.ru                    ">
            quality_control = <"British Medical Translator id 00400595">
        >
        ["ru"] = <
            provenance = <"gost.e-ecolog.ru                    ">
            quality_control = <"Russion Translator id 892230A">
>
```

Архетипы всегда должны переводиться полностью, или не переводиться вообще. Это означает, что при новом переводе каждый зависящий от языка раздел секций description и ontology должен быть переведен на новый язык, и соответствующее дополнение должно быть внесено в список translations в секции language.

8.5.3.6 Секция description (описание)

Секция description архетипа содержит описательную информацию (иногда называемую метаданными документа), например элементы, которые могут использоваться в индексах фонда архетипов и для поиска. Для ее описания используется синтаксис языка dADL.

Пример

```
description
    original_author = <
    ["name"] = <"Dr J Joyce">
    ["organisation"] = <"NT Health Service">
    ["date"] = <2003-08-03>
>
lifecycle_state =                   <"initial">
archetype_package_uri =
    <"gost.e-ecolog.ru                                    >
details = <
    ["en"] = <
        purpose = <"archetype for diabetic patient review">
        use = <"used for all hospital or clinic-based diabetic reviews,
            including first time. Optional sections are removed according
```

```
            to the particular review">
        misuse = <"not appropriate for pre-diagnosis use">
        original_resource_uri =
            <"www.healthdata.org.au/data_sets/
                                diabetic_review_data_set_1.html">
        other_details = <...>
    >
    ["de"] = <
        purpose = <"Archetyp für die Untersuchung von Patienten
                mit Diabetes">
        use = <"wird benutzt für alle Diabetes-Untersuchungen im
            Krankenhaus, inklusive der ersten Vorstellung. Optionale
            Abschnitte werden in Abhängigkeit von der speziellen
            Vorstellung entfernt.">
        >
        misuse = <"nicht geeignet für Benutzung vor Diagnosestellung">
        original_resource_uri =
            <"www.healthdata.org.au/data_sets/
                                diabetic_review_data_set_1.html">
        other_details =          <...>
    >
>
```

8.5.4 Секция definition (определение)

Секция definition содержит основное формальное определение архетипа, записанное на языке определения ограничений (cADL).

Пример

```
definition
    ENTRY[at0000] ∈ {                                   - - измерение кровяного давления
        name ∈ {                                        - - любой синоним КД
            CODED_TEXT ∈ {
                code ∈ {
                    CODE_PHRASE ∈ {[ac0001]}
                }
            }
        }
        data ∈ {
            HISTORY[at9001] ∈ {                                 - - история
                events cardinality ∈ {1..*} ∈ {
                    EVENT[at9002] occurrences ∈ {0..1} ∈ {          - - базовый уровень
                        name ∈ {
                            CODED_TEXT ∈ {
                                code ∈ {
                                    CODE_PHRASE ∈ {[ac0002]}
                                }
                            }
                        }
                        data ∈ {
                            LIST_S[at1000] ∈ {                      - - общее артериальное КД
                                items cardinality ∈ {2..*} ∈ {
                                    ELEMENT[at1100] ∈ {             - - систолическое КД
                                        name ∈ {                    - - любой синоним «систолическое»
                                            CODED_TEXT ∈ {
                                                code ∈ {
                                                    CODE_PHRASE ∈ {[ac0002]}
                                                }
                                            }
                                        }
                                        value ∈ {
                                            QUANTITY ∈ {
                                                magnitude ∈ {0..1000}
```

```
                        property ∈ {[properties::0944]} - - "давление"
                        units ∈ {[units::387]} - - "мм [рт.ст.]"
                    }
                }
            }
            ELEMENT[at1200] ∈ {                 - - диастолическое КД
                name ∈ {                        - - любой синоним «диастолическое»
                    CODED_TEXT ∈ {
                        code ∈ {
                          CODE_PHRASE ∈ {[ac0003]}
                        }
                    }
                    }
                value ∈ {
                    QUANTITY ∈ {
                        magnitude ∈ {0..1000}
                        property ∈ {[properties::0944]} - - "давление"
                        units ∈ {[units::387]} - - "мм [рт.ст.]"
                    }
                }
            }
            ELEMENT[at9000] occurrences ∈ {0..*} ∈ {*}
                                        - - неизвестный новый элемент
        }
    ...
```

8.5.5 Секция invariant (инвариант)

В архетипе на языке ADL секция invariant содержит утверждения, относящиеся ко всему архетипу, и может использоваться для создания формулировок, которые невозможны в блочной структуре секции definition. К данной категории относится любое ограничение, которое связывает более чем одно свойство с другим, а также большинство ограничений, содержащих математические или логические формулы. Формулировка инварианта является формулировкой логики предикатов первого порядка, которая может быть приведена к булевскому результату во время исполнения. Ссылки на объекты и свойства осуществляются с использованием путей.

Пример

```
invariant
    validity: /[at0001]/speed[at0002]/kilometres/magnitude =
        /[at0003]/speed[at0004]/miles/magnitude * 1.6
```

8.5.6 Секция ontology (онтология)

8.5.6.1 Обзор

Секция архетипа ontology записывается на языке dADL. В ней определяются коды, представляющие идентификаторы узлов, ограничения текста или терминов и связи с терминологиями. Лингвистические переводы добавляются в виде дополнительных блоков, помеченных соответствующим языком.

Пример

```
ontology
    terminologies_available = <"snomed_ct", ...>
    term_definitions = <
        ["en"] = <
            items = <...>
        >
        ["de"] = <
            items = <...>
        >
    >
    term_binding = <
        ["snomed_ct"] = <
            items = <...>
        >
        ...
    >
```

```
    constraint_definitions = <
        ["en"] = <...>
        ["de"] = <
            items = <...>
        >
        ...
    >
    constraint_binding = <
        ["snomed_ct"] = <...>
        ...
    >
```

Секция определения терминов term_definitions является обязательной и должна быть указана для каждого выполненного перевода.

Каждая из этих секций может иметь свои метаданные, которые появляются в подсекциях описаний, наподобие показанной выше, содержащей детали перевода.

8.5.6.2 Объявления в заголовке секции ontology

Объявление terminologies_available содержит идентификаторы всех терминологий, для которых указаны секции term_binding.

8.5.6.3 Секция term_definitions (определения терминов)

Эта секция расположена там, где определены все местные термины архетипа (т.е. термины в форме [atNNNN]). Каждый термин определяется с использованием структуры пар имя-значение, и должен содержать, по крайней мере, имена "text" и "description". Затем каждый объект, представляющий термин, включается в список определений терминов на соответствующем языке.

Примечание — Следующий пример показывает выдержку из определений местных терминов на английском и немецком языках , указанных в архетипе заголовков проблемно-ориентированной истории болезни (SOAP):

```
    term_definitions = <
        ["en"] = <
            items = <
                ["at0000"] = <
                    text = <"problem">
                    description = <"The problem experienced by the subject of care
                                    to which the contained information relates">
                >
                ["at0001"] = <
                    text = <"problem/SOAP headings">
                    description = <"SOAP heading structure for multiple problems">
                >
                ...
                ["at4000"] = <
                    text = <"plan">
                    description = <"The clinician's professional advice">
                >
            >
        >
        ["de"] = <
            items = <
                ["at0000"] = <
                    text = <"klinisches Problem">
                    description = <"Das Problem des Patienten worauf sich diese \ Informationen beziehen">
                >
                ["at0001"] = <
                    text = <"Problem/SOAP Schema">
                    description = <"SOAP-Schlagwort-Gruppierungsschema fuer mehrfache Probleme">
                >
                ["at4000"] = <
                    text = <"Plan">
                    description = <"Klinisch-professionelle Beratung des Pflegenden">
                >
            >
        >
    >
```

Метка происхождения provenance может использоваться для указания источника определений терминов.

Пример

```
[at4000] = <
    text = <"plan">;
    description = <"The clinician's professional advice">;
    provenance = <"ACME_terminology(v3.9a)">
>
```

Примечание — В данном примере показано только происхождение термина, а не его связь с каким-либо другим термином. Связи описаны в подразделах 8.5.6.5 и 8.5.6.6.

8.5.6.4 Секция constraint_definitions (определения ограничений)

Секция constraint_definitions имеет точно такую же форму, что и секция term_definitions, и тоже содержит определения, т. е. смысловые значения кодов местных ограничений, имеющих вид [acNNNN]. Определения ограничений не содержат формулировки ограничений, а только описывают их назначение. Формулировки ограничений содержатся в секции constraint_binding.

Пример

```
items = <
    [ac1015] = <
        text = <"type of hepatitis">
        description = <"any term which means a kind of viral hepatitis">
    >
>
```

8.5.6.5 Секция term_binding (связь терминов)

Данная секция используется для описания эквивалентности местных терминов архетипа и терминов из внешних терминологий. Каждое отображающее выражение указывает, какой термин из внешней терминологии эквивалентен внутренним кодам архетипа.

Пример

```
term_binding(umls) = <
    [umls] = <
        items =<
            [at0000] = <[umls::C124305]> - - результат оценки по шкале Апгар
            [at0002] = <[umls::0000000]>  - - оценка 1-й минуты по шкале Апгар
            [at0004] = <[umls::C234305]> - - оценка сердцебиения
            [at0005] = <[umls::C232405]> - - оценка дыхания
            [at0006] = <[umls::C254305]> - - оценка мышечного тонуса
            [at0007] = <[umls::C987305]> - - оценка рефлекторной возбудимости
            [at0008] = <[umls::C189305]> - - оценка цвета кожи
            [at0009] = <[umls::C187305]> - - оценка по шкале Апгар
            [at0010] = <[umls::C325305]> - - оценка 2-й минуты по шкале Апгар
            [at0011] = <[umls::C725354]> - - оценка 5-й минуты по шкале Апгар
            [at0012] = <[umls::C224305]> - - оценка 10-й минуты по шкале Апгар
        >
    >
>
```

8.5.6.6 Секция constraint_binding (связь ограничений)

В данной секции основного тела архетипа формально описаны текстовые ограничения. Они описаны отдельно, так как зависят от терминологии и у конкретного логического ограничения таких описаний может быть несколько.

Пример

```
constraint_binding = <
    ["snomed_ct"]
        items = <
```

```
            ["ac0001"] = <gost.e-ecolog.ru                terminology_id=snomed_ct&&
                                has_relation=[102002];with_target=[128004]>
            ["ac0002"] = <gost.e-ecolog.ru
                                gost.e-ecolog.ru                >
        >
    >
>
```

8.5.7 Секция revision_history (история версий)

Секция revision_history архетипа описывает аудиторскую историю изменений архетипа с использованием синтаксиса языка dADL. Данная секция является необязательной и добавляется в конец определения архетипа.

Пример

```
revision_history
revision_history = <
    ["1.57"] = <
        committer = <"Miriam Hanoosh">
        committer_organisation = <"AIHW.org.au">
        time_committed = <2004-11-02 09:31:04+1000>
        revision = <"1.2">
        reason = <"Added social history section">
        change_type = <"Modification">
    >
    - - и т. д.
    ["1.1"] = <
        committer = <"Enrico Barrios">
        committer_organisation = <"AIHW.org.au">
        time_committed = <2004-09-24 11:57:00+1000>
        revision = <"1.1">
        reason = <"Updated HbA1C test result reference">
        change_type = <"Modification">
    >
    ["1.0"] = <
        committer = <"Enrico Barrios">
        committer_organisation = <"AIHW.org.au">
        time_committed = <2004-09-14 16:05:00+1000>
        revision = <"1.0">
        reason = <"Initial Writing">
        change_type = <"Creation">
    >
>
```

8.5.8 Проверка правильности

8.5.8.1 Общая информация

Ниже описана формальная (т. е. проверяемая) семантика архетипов на языке ADL.

8.5.8.2 Глобальная правильность архетипа

Следующие ограничения правильности применяются к архетипу в целом.

Примечание — Далее термин «секция» означает то же самое, что и «атрибут», т. е. секция, называемая «определением» в тексте на языке dADL, является сериализацией значения атрибута с тем же именем.

VARID: archetype identifier validity (правильность идентификатора архетипа). Архетип должен иметь значение идентификатора в секции archetype_id.

VARCN: archetype concept validity (правильность понятия архетипа). Архетип должен иметь значение термина архетипа в секции concept. Данный термин должен существовать в онтологии архетипа.

VARDF: archetype definition validity (правильность определения архетипа). Архетип должен иметь секцию definition, представленную в виде строк на языке cADL или с эквивалентным подключаемым синтаксисом.

VARON: archetype ontology validity (правильность онтологии архетипа). Архетип должен иметь секцию ontology, представленную в виде строк на языке cADL или с эквивалентным подключаемым синтаксисом.

VARDT: archetype definition typename validity (правильность имени типа данных в определении архетипа). Самое верхнее имя типа данных, упомянутое в секции definition архетипа, должно соответствовать типу данных, упомянутому в слоте имени типа данных в первом сегменте секции archetype_id.

8.5.8.3 Правильность кодированных терминов

Все идентификаторы узлов (коды, начинающиеся с "at"), используемые в секции definition архетипа, должны быть определены в секции term_definitions онтологии.

VATDF: archetype term validity (правильность терминов архетипа). Каждый термин архетипа, используемый как идентификатор узла в определении архетипа, должен быть определен в секции term_definitions онтологии.

Все идентификаторы ограничений (коды, начинающиеся с "ac"), используемые в секции definition архетипа, должны быть определены в секции constraint_definitions онтологии.

VACDF: node identifier validity (правильность идентификатора узла). Каждый код ограничения, используемый в определении архетипа, должен быть определен в секции constraint_definitions онтологии.

8.5.8.4 Секция definition (определение)

Следующее ограничение применяется к секции definition архетипа.

VDFPT: path validity in definition (правильность пути в определении). Любой путь, упомянутый в секции definition, должен быть синтаксически правильным и иметь правильное значение по отношению к иерархической структуре секции definition.

8.5.9 Синтаксис архетипа

8.5.9.1 Грамматика

input:
  archetype
| error

archetype:
  arch_identification arch_specialisation arch_concept arch_description
arch_definition arch_invariant arch_ontology

arch_identification:
  arch_head V_ARCHETYPE_ID
| SYM_ARCHETYPE error

arch_head:
  SYM_ARCHETYPE
| SYM_ARCHETYPE arch_meta_data

arch_meta_data:
  (arch_meta_data_items)

arch_meta_data_items:
  arch_meta_data_item
| arch_meta_data_items ; arch_meta_data_item

arch_meta_data_item:
  SYM_ADL_VERSION = V_VERSION_STRING
| SYM_IS_CONTROLLED

arch_specialisation:
  -/-
| SYM_SPECIALIZE V_ARCHETYPE_ID
| SYM_SPECIALIZE error

arch_concept:
  SYM_CONCEPT V_LOCAL_TERM_CODE_REF
| SYM_CONCEPT error

arch_description:
  -/-
| SYM_DESCRIPTION V_DADL_TEXT
| SYM_DESCRIPTION error

arch_definition:
  SYM_DEFINITION V_CADL_TEXT
| SYM_DEFINITION error

```
    arch_invariant:
      -/-
    |  SYM_INVARIANT V_ASSERTION_TEXT
    |  SYM_INVARIANT error
    arch_ontology:
      SYM_ONTOLOGY V_DADL_TEXT
    |  SYM_ONTOLOGY error
```

8.5.9.2 Символы

```
---------- /* символы */ ----------------------------------------------------------------
"-"          Minus_code
"+"          Plus_code
"*"          Star_code
"/"          Slash_code
"^"          Caret_code
"="          Equal_code
"."          Dot_code
";"          Semicolon_code
","          Comma_code
":"          Colon_code
"!"          Exclamation_code
"("          Left_parenthesis_code
")"          Right_parenthesis_code
"$"          Dollar_code
"?"          Question_mark_code
"["          Left_bracket_code
"]"          Right_bracket_code

---------- /* ключевые слова */ ----------------------------------------------------------
^[Aa][Rr][Cc][Hh][Ee][Tt][Yy][Pp][Ee][ \t\r]*\n                SYM_ARCHETYPE
^[Ss][Pp][Ee][Cc][Ii][Aa][Ll][Ii][SsZz][Ee][  \t\r]*\n         SYM_SPECIALIZE
^[Cc][Oo][Nn][Cc][Ee][Pp][Tt][ \t\r]*\n                        SYM_CONCEPT
^[Dd][Ee][Ff][Ii][Nn][Ii][Tt][Ii][Oo][Nn][ \t\r]*\n            SYM_DEFINITION
 - - мини-анализатор для генерации V_DADL_TEXT

^[Dd][Ee][Ss][Cc][Rr][Ii][Pp][Tt][Ii][Oo][Nn][ \t\r]*\n        SYM_DESCRIPTION
 - - мини-анализатор для генерации V_CADL_TEXT
^[Ii][Nn][Vv][Aa][Rr][Ii][Aa][Nn][Tt][ \t\r]*\n                 SYM_INVARIANT
- - мини-анализатор для генерации V_ASSERTION_TEXT
^[Oo][Nn][Tt][Oo][Ll][Oo][Gg][Yy][ \t\r]*\n                    SYM_ONTOLOGY
- - мини-анализатор для генерации V_DADL_TEXT
---------- /* ссылка на коды терминов */ ---------------------------------------------------
\[[a–zA–Z0–9][a–zA–Z0–9.–]*\]                                  V_LOCAL_TERM_CODE_REF
---------- /* идентификатор архетипа */ ----------------------------------------------------
[a–zA–Z][a–zA–Z0-9_–]+\.[a–zA–Z][a–zA–Z0—9_–]+\.[a–zA–Z0–9]+
                                                               V_ARCHETYPE_ID
---------- /* идентификаторы */ ------------------------------------------------------------
[a–zA–Z][a–zA–Z0–9_]*                                          V_IDENTIFIER
```

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 639	—	*
ИСО 8601	IDT	ГОСТ ИСО 8601—2001 «Система стандартов по информации, библиотечному и издательскому делу. Представление дат и времени. Общие требования»
ИСО/МЭК 10646	—	*

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

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

- IDT — идентичные стандарты.

Библиография

[1] EN 13940-1, Health informatics — System of concepts to support continuity of care — Part 1: Basic concepts
[2] CEN/TS 14796, Health informatics — Data types
[3] ISO 3166 (all parts), Codes for the representation of names of countries and their subdivisions
[4] ISO/IEC 10746-1:1998, Information technology — Open distributed processing — Reference model: Overview — Part 1
[5] EN 14822-2:2005, Health informatics — General purpose information components — Part 2: Nonclinical
[6] ISO 1087-1:2000, Terminology work — Vocabulary — Part 1: Theory and application
[7] ISO/IEC 11179-3:2003, Information technology — Metadata registries (MDR) — Part 3: Registry metamodel and basic attributes
[8] ISO/IEC 11404, Information technology — General-Purpose Datatypes (GPD)
[9] ISO/TS 18308:2004, Health informatics — Requirements for an electronic health record architecture
[10] ISO/TR 20514:2005, Health informatics — Electronic health record — Definition, scope and context
[11] RFC 1738:2004, Uniform Resource Locators (URL)
[12] RFC 2045:1996, Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies
[13] RFC 2046:1996, Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types
[14] RFC 2806:2000, URLs for Telephone Calls
[15] RFC 2936:2000, HTTP MIME Type Handler Detection
[16] RFC 2978:2000, IANA Charset Registration Procedures

Научно-исследовательские проекты, давшие информацию для подхода на основе архетипов:

[17] Moorman, P.W., van Ginneken, A.M., van der Lei, J. and van Bemmel, J.H., A model for structured data entry based on explicit descriptional knowledge, Methods of Information in Medicine, 33(5), pp 454—63, December 1994
[18] Dore, L., Lavril, M., Jean, F.C. and Degoulet P.A., Development environment to create medical applications, GREENES, R.A. et al., eds., Medinfo, 8, pp 185—189, 1995
[19] Kalra D., ed., Synapses ODP Information Viewpoint, EU Telematics Application Programme, Brussels, 1998; The Synapses Project: Final Deliverable, 10 chapters, 64 pages
[20] Beale T. The GEHR Archetype System, The Good Electronic Health Record Project, Australia, August 2000 http://www.openehr.org/downloads/usage/gehr_australia/gehr_archetypes.pdf

Публикации о подходе на основе архетипов

[21] Kalra, D., Clinical Foundations and Information Architecture for the Implementation of a Federated Health Record Service, PhD Thesis, University of London, 2002. Available from:

gost.e-ecolog.ru

[22] Beale, T., Archetypes — An Interoperable Knowledge Methodology for Future-proof Information Systems. 2001, 69 pages. Available from: BEALE, T. Archetypes: Constraint-Based Domain Models for Future-proof Information Systems, in OOPSLA-2002, Workshop on behavioural semantics, 2002
[23] The openEHR Foundation: Archetypes FAQ. Available from: http://www.openehr.org/shared-resources/faqs/archetypes.html

Другая современная работа выполняется HL7 Templates SIG и HL7 Care Provision Technical Committee.

Проводится исследование по переводу ограничений архетипов с языка ADL на OWL. OWL-представление архетипов может в будущем играть дополняющую роль по отношению к ADL-представлению

УДК 61:004:006.354 ОКС 35.240.80 П85 ОКСТУ 4002

Ключевые слова: здравоохранение, информатизация здоровья, электронные медицинские карты, передача электронных медицинских карт, архетипы, передача архетипов, модели архетипов

Редактор *В. Л. Савинова*
Технический редактор *В. Н. Прусакова*
Корректор *Л. Я. Митрофанова*
Компьютерная верстка *В. Н. Романовой*

Сдано в набор 04.10.2013. Подписано в печать 30.01.2014. Формат 60×84 1/8. Бумага офсетная. Гарнитура Ариал. Печать офсетная. Усл. печ. л. 13,02. Уч.-изд. л. 11,95. Тираж 55 экз. Зак. 1455.

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

gost.e-ecolog.ru