

Московский государственный университет имени М.В. Ломоносова
факультет вычислительной математики и кибернетики
кафедра системного программирования



Дипломная работа

**Интеграция данных по свойствам веществ в специализированное
пространство связанных данных**

Выполнила:
Студентка 528 группы
Устинова Е.С.

Научный руководитель:
проф. д.ф.-м.н. Серебряков В.А.

Москва, 2014

Аннотация

Данная работа проводилась в рамках исследований [1,2], посвященных интеграции данных по свойствам веществ в пространство Linked Open Data.

На основе результатов работы [2] была разработана онтология предметной области с использованием терминов уже существующих онтологий смежных предметных областей (QUDT [3], ChemAxiom [4]). Дополнительно были проанализированы и промоделированы ограничения предметной области. В том числе был разработан подход к моделированию ограничений на численные значения свойств веществ на основе OWL. В качестве реализации данного подхода был создан веб-интерфейс для определения указанных ограничений.

Исходные данные набора, представленные в реляционном виде, были преобразованы в формат RDF с помощью инструмента D2R [5]. К полученным RDF-данным с помощью D2R был предоставлен доступ в соответствии с рекомендациями Linked Data. Также был реализован интерфейс для связывания ресурсов-веществ из полученного набора RDF-данных с ресурсами-веществами набора ChemSpider [6].

Оглавление

Введение.....	5
Постановка задачи.....	9
Глава 1. Обзор возможностей RDFS и OWL моделирования.....	10
Глава 2. Обзор тематически близких источников данных и онтологий Linked Open Data.....	14
2.1 Представление химических данных в Linked Open Data.....	14
2.1.1 Краткое описание распространенных химических онтологий.....	15
2.1.2 Источники данных о веществах.....	16
2.2 Измеряемые свойства и единицы измерения в Linked Open Data.....	22
2.3 Общие выводы из обзора тематически схожих источников данных и онтологий.....	26
Глава 3. Подход к интеграции данных по свойствам веществ в пространство Linked Open Data.....	28
3.1 Построение онтологии предметной области.....	28
3.1.1 Повторное использование терминов из существующих онтологий.....	28
3.1.2 Анализ ограничений предметной области.....	35
3.2 Преобразование исходных данных в RDF.....	39
3.3 Подход к связыванию данных по свойствам веществ с внешними источниками.....	39
3.3.1 Связывание с внешними источниками по веществам.....	40
3.3.2 Связывание с внешними источниками по свойствам и единицам измерения и задание ограничений на численные значения свойств.....	41
Глава 4. Реализация интеграции данных по свойствам веществ в пространство Linked Open Data.....	44
4.1 Модель ограничений предметной области в разрабатываемой онтологии.....	44
4.1.1 Непересекаемость классов свойств-функций и свойств-констант.....	44
4.1.2 Согласованность ссылок на состояния вещества при определении наборов данных, фрагментов данных и применимости свойств-функций к состояниям.....	45
4.2 Создание новых экземпляров-свойств и единиц измерения в терминах QUDT и	

задание ограничений на численные значения свойств.....	49
4.2.1 Реализованные интерфейсы.....	49
4.2.2 Принцип работы создаваемых ограничений на значения свойств.....	54
4.3 Реализация связывания сущностей-веществ из исходных данных с ресурсами	
ChemSpider.....	54
Заключение.....	57
Используемые термины.....	58
Список литературы.....	60
Приложение А. Схема исходной реляционной базы данных.....	62
Приложение В. Общая схема разработанной онтологии.....	63
Приложение С. Иерархия подклассов Data.....	64
Приложение D. Иерархия подклассов Dataset.....	65
Приложение Е. Схема представления ограничений на численные значения свойств (определенных в терминах онтологии QUDT), записанных на языке OWL.....	66

Введение

Данная работа продолжает цикл исследований, посвященных Semantic Web и, в частности, методам онтологического моделирования применительно к тематике «теплофизические свойства веществ» [1,2]. В предыдущих работах коллектива авторов обоснован выбор технологий Semantic Web как наиболее перспективного подхода к интеграции разнородных научных данных и детализированы требования к построению онтологии – формализованной системе понятий предметной области, которая представлена как набор сущностей, соединенных различными отношениями.

Целью данной работы являлась интеграция данных по теплофизическим свойствам веществ, предоставленных ОИВТ РАН (Объединенный институт высоких температур РАН), в пространство Открытых Связанных Данных (далее Linked Open Data).

Задача интеграции должна решаться отдельно для каждого отдельно взятого набора данных, так как процесс ее решения не является строго зафиксированным и может различаться для разных исходных данных и предметных областей. Однако можно выделить основные этапы интеграции : 1) разработка онтологии предметной области или выбор одной из существующих онтологий, 2) получение необходимых данных в RDF формате, соответствующем онтологии предметной области, 3) предоставление доступа к полученным данным в соответствии требованиями Linked Open Data, 4) генерация типизированных связей между ресурсами полученного набора данных и ресурсами внешних наборов в Linked Open Data.

Важность тщательного выполнения интеграции может быть проиллюстрирована проблемами, которые в данный момент существуют в Linked Open Data: проблема разнообразия терминов и разобщенность данных. Под проблемой разнообразия терминов понимается ситуация, когда для описания одних и тех же понятий (классов) одной и той же предметной области в различных наборах данных используются различные онтологии. Такая ситуация в целом противоречит концепции данных, пригодных для машинной обработки. В частности, это может привести, например, к невозможности обеспечить однородный доступ к данным из таких наборов. Под разобщенностью данных понимается отсутствие ссылок на ресурсы, определенные в других наборах данных, что, например, приводит к большому числу дублирующих ресурсов. В качестве примера можно привести ситуацию, когда в различных источниках описывается одно и то же явление или объект, но при этом между этими описаниями

нет связи, которая бы идентифицировала что действительно описано одно и то же.

Перечисленные проблемы являются следствием нестандартизованности процесса интеграции набора данных в пространство Linked Open Data.

Итак, первостепенной задачей, возникающей в процессе интеграции в пространство Linked Open Data, является построение онтологии предметной области.

Главный объект описания для рассматриваемого набора данных - численные значения теплофизических свойств для различных веществ в разных условиях среды. Основные

Hydroxyl OH(g)

$\Delta_f H_0^\circ$	39.110 kJ/mol	$\Delta_f H_{298}^\circ$	39.349 kJ/mol
----------------------	---------------	--------------------------	---------------

T K	Cp J/K*mol	F J/K*mol	S J/K*mol	H kJ/mol
298.15	29.886	154.068	183.627	8.813
300.00	29.879	154.251	183.812	8.868
400.00	29.606	162.763	192.366	11.841
500.00	29.487	169.368	198.957	14.795
600.00	29.512	174.761	204.333	17.743
700.00	29.663	179.319	208.892	20.701
800.00	29.921	183.270	212.869	23.679
900.00	30.265	186.759	216.412	26.688
1000.00	30.674	189.887	219.622	29.735
1100.00	31.126	192.726	222.566	32.824
1200.00	31.601	195.328	225.295	35.961
1300.00	32.077	197.732	227.843	39.145
1400.00	32.533	199.969	230.237	42.375
1500.00	32.947	202.063	232.496	45.650
1500.00	32.904	202.063	232.494	45.646
1600.00	33.370	204.032	234.632	48.960
1700.00	33.781	205.893	236.668	52.318
1800.00	34.149	207.657	238.609	55.715

Рис. 1 Фрагмент исходных данных

сущности данной предметной области: вещества, свойства, единицы измерения, наборы численных значений свойств для вещества в определенном состоянии, численные значения свойств. Вещества характеризуются систематическим названием, молекулярной формулой и могут находиться в различных фазовых состояниях. Фазовые состояния включают в себя набор агрегатных состояний (газ, жидкость, твердое состояние), а также состояния насыщения (газ-жидкость, газ-твердое, жидкость-твердое) и тройную точку (газ-жидкость-твердое), отдельно выделяется состояние

идеального газа. Для каждого свойства указывается название, сокращенное обозначение и тип. По типу свойства делятся на свойства-функции и свойства-константы. Для свойств-функций определяется набор аргументов (другие свойства, как правило это температура и/или давление), область определения, область значений, может быть указан характер монотонности. Числовые значения свойств-функций задаются для определенных значений их аргументов. Свойства-константы не имеют аргументов. Их числовые значения задаются для вещества без указания дополнительных параметров. Свойства могут быть применимы к одним состояниям веществ и не применимы к другим. Числовые значения свойств даются с указанием единиц измерения, а также для каждого числового значения как правило указывается источник в виде библиографической записи (например, справочник теплофизических свойств веществ).

На рис. 1 представлен фрагмент исходных данных: для вещества Hydroxyl с формулой OH в состоянии газа заданы значения свойств на сетке температур. Самый левый столбик "Т" — сетка температур, в остальных — значения свойств, зависящих от температуры. Наверху под заголовком заданы значения константных свойств для данного вещества. В данном примере показана ситуация для газового состояния, однако ситуация несколько меняется, если фазовое состояние вещества включает в себя несколько агрегатных состояний, например, состояние «газ-жидкость». В таком случае свойства, применимые к обоим отдельным состояниям газ и жидкость, будут иметь разные значения для «подсостояния» газа и «подсостояния» жидкости. Некоторые свойства применимы только лишь для состояний насыщения, для них числовые значения должны быть указаны один раз (вне зависимости от «подсостояния» вещества). Свойства для состояния вещества в тройной точке не рассматриваются.

В работе [2] были введены основные классы и связи, необходимые для моделирования рассматриваемой предметной области, однако необходимо, учитывая описанные выше проблемы Linked Open Data, рассмотреть возможности повторного использования классов и связей из уже существующих онтологий. Такое повторное использование способствует интеграции, а также может улучшить структуру RDF-данных. В данный момент существует множество онтологий, которые частично покрывают рассматриваемую предметную область, однако не было обнаружено онтологий в свободном доступе, которые были бы пригодны для полного представления рассматриваемых данных в RDF.

Одним из главных назначений онтологий в Semantic Web является моделирование ограничений, накладываемых предметной областью на данные. В работе [2] такие ограничения рассмотрены не были, поэтому их анализ и моделирование также являлось целью данной работы. На основе результатов [2] была разработана онтология предметной области с использованием терминов онтологий QUDT (область: численные свойства и их значения, а также единицы измерения) и ChemAxiom (область: данные о веществах). Исходные данные набора, представленные в реляционном виде, были преобразованы в формат RDF с помощью инструмента D2R. К полученным RDF-данным с помощью D2R был предоставлен доступ в соответствии с рекомендациями Linked Data, сформулированными в [7]. Был реализован интерфейс для связывания ресурсов-веществ из полученного набора RDF-данных с ресурсами-веществами набора ChemSpider. Был разработан подход к моделированию ограничений на численные значения свойств веществ на основе OWL-моделирования. Был реализован

интерфейс для определения указанных ограничений.

Постановка задачи

Требуется провести интеграцию в пространство Linked Open Data данных о теплофизических свойствах веществ.

Исходные данные хранятся в реляционной базе данных, схема которой уже представляет некоторую формализацию предметной области, данная схема соответствует онтологии, описанной в [2].

Требуется найти и изучить, онтологии, уже разработанные для данной предметной области, и выбрать из них подходящие для использования, учитывая основные классы и связи, введенные в [2], составить OWL-онтологию предметной области с использованием терминов выбранных онтологий. Разработанная онтология должна отображать ограничения, задаваемые предметной областью, выраженные на языке OWL.

Требуется с помощью существующих инструментов публикации преобразовать исходные реляционные данные в формат RDF, соответствующий разработанной онтологии. Таким образом, на онтологию накладывается дополнительное требование : исходные данные должны быть представимы в виде, соответствующем онтологии.

Также требуется сделать обзор по существующим наборам данных в Linked Open Data и найти наборы, пригодные для связывания с нашими данными, определить способ связывания и реализовать его (это может быть, например, автоматическая генерация связей по определенным правилам). Требуется рассмотреть возможность извлечения из внешних источников дополнительных полезных данных об объектах предметной области (к таким дополнительным данным могут относиться данные, задающие некоторым образом ограничения предметной области, но при этом отсутствующие в нашем исходном наборе).

Глава 1. Обзор возможностей OWL и RDFS моделирования

RDF позволяет описывать данные о предметах или явлениях, идентифицированных с помощью URI, используя форму тройки или триплета : субъект—предикат—объект. Субъект — это идентифицированный с помощью URI ресурс, обозначающий какой либо предмет или явление, предикат — типизированная связь, обозначающая свойства, которым обладает рассматриваемый ресурс, объект — это ресурс или литерал, который представляет собой значение свойства, обозначенного предикатом тройки.

Примером такого представления данных может служить тройка «Небо — имеет цвет—синий». Здесь субъект — «Небо» обладает свойством «имеет цвет», обозначенным соответствующим предикатом, значением данного свойства является «синий цвет». Отметим, что в данном примере объект рассмотренного отношения может быть как литералом (если мы значение свойства «имеет цвет» запишем обычной строкой), так и ресурсом (если мы для обозначения каждого цвета заведем ресурс с URI и будем использовать его). Уже на данном примере видно, что предикаты по типу можно разделить на литеральные и объектные, что и делается в OWL.

RDF предоставляет лишь определенную форму представления данных, а также некоторые основные термины : классы и предикаты. Основным предикатом, введенным RDF, является `rdf:type`, с помощью него можно обозначать тип ресурса. Для описания самих типов (или классов) ресурсов и типов связей используются языки RDFS и OWL, основанные на RDF. Эти языки позволяют в форме RDF описать классы и связи предметной области, а также некоторые ограничения на использование этих классов и связей.

RDFS является подмножеством OWL, определяя основные типы объектов онтологий : `rdfs:Class` (класс), `rdfs:Property`(свойство или предикат); а также предикаты, для установления связей между объектами данных типов : `rdfs:subClassOf`, `rdfs:subPropertyOf`. Смысл каждого из вводимых типов и предикатов определяется с помощью указания логического вывода [8], который следует из RDF-троек, в которых участвуют указанные типы и предикаты. Таким образом мы подошли к понятию логического вывода в RDFS и OWL. Следует отметить, что существуют записанные тройки (англ. *asserted triples*) – тройки явно указанные в наборе; и выведенные тройки (англ. *inferred triples*) – тройки, выводимые из троек заданного набора с помощью логического вывода. Например, `rdfs:subClassOf` имеет следующее определение [9]:

Если есть записанная тройка : «Aclass – class rdfs:subClassOf – Bclass», то по определению предиката rdfs:subClassOf делается вывод, что тип ресурсов Aclass и Bclass – это rdfs:Class. При этом, если есть еще записанная тройка «Aobject – rdf:type – Aclass», то отсюда выводится тройка : «Aobject – rdf:type – Bclass». То есть все объекты класса Aclass являются также объектами класса Bclass.

Как видно из приведенных примеров конструкций, уже на уровне RDFS все ресурсы, исключая предикаты, делятся на две группы : классы и экземпляры классов(instance, individual). Классы это ресурсы, для которых rdf:type имеет значение rdfs:Class, экземпляры классов — это ресурсы, для которых trd:type имеет значение какого-то заранее определенного класса.

Также RDFS вводит термины rdfs:domain и rdfs:range, которые служат для обозначения типов на концах связей, задаваемых RDF-тройками. Например, если предикат «любит на обед» имеет класс «Животное» в качестве rdfs:domain, и класс «Еда» в качестве rdfs:range, то из тройки «Кот_Вася – любит на обед – колбаса» выводятся следующие тройки : «Кот_Вася – rdf:type – животное», «Колбаса – rdf:type – Еда».

OWL вводит дополнительные возможности моделирования [8, 10]. Вводятся предикаты для обозначения типа предикатов : owl:ObjectProperty (объектные свойства, в качестве объекта отношения - ресурс) и owl:DatatypeProperty (литеральные свойства, в качестве объекта отношения — литеральное значение). В OWL также вводятся дополнительные виды предикатов, обладающих различными свойствами (например, транзитивность, симметричность, идентифицирующие предикаты, предикаты, инверсные по отношению к другим). Вводятся дополнительные способы задания ограничений на типы ресурсов-объектов отношений. С помощью конструкции Restriction можно задавать различные ограничения. Такие ограничения могут касаться количества триплетов, для ресурса заданного типа с определенным предикатом. Может быть указано их минимальное и максимальное количество. Например, нужно промоделировать, что объект типа «Книга» должен иметь как минимум одного автора. Запишем требуемую модель с помощью одной из нотаций RDF : n-triple.

```
Книга rdf:type owl:Class.  
Книга rdfs:subClassOf Restriction [  
    owl:onProperty имеетАвтора.  
    owl:minCardinality 1.  
]
```

Конструкция Restriction также позволяет накладывать некоторые ограничения на используемые типы. Например, если нужно, чтобы автором книги обязательно являлся человек, то мы запишем:

```
Книга rdf:type owl:Class.  
Человек rdf:type owl:Class.  
Книга rdfs:subClassOf Restriction [  
    owl:onProperty имеетАвтора.  
    owl:AllValuesFrom Человек.  
]
```

Формально это означает, что любой экземпляр, являющийся объектом отношения, задаваемого предикатом «имеетАвтора» для субъекта-экземпляра класса «Книга», будет отнесен к классу «Человек». Существуют также и другие типы ограничений (owl:someValuesFrom – хотя бы один объект предиката существует, и при этом его класс задан; owl:hasValue – объектом заданного предиката является заданный конкретный экземпляр).

Также OWL предусматривает операции с множествами для классов. Например, если некоторый класс А является подклассом классов В и С. То экземпляры класса А принадлежат пересечению множества экземпляров класса и класса В. Для того же, чтобы обозначить, что класс А совпадает с пересечением классов В и С, нужны дополнительные средства выразительности. Итак, OWL определяет термины для обозначения классов-пересечений и классов-объединений. В OWL есть особые предикаты, с помощью которых можно указать, что заданные два класса не могут иметь общих экземпляров (owl:disjointWith), или же что наоборот, множества экземпляров заданных классов полностью совпадают (owl:equivalentClass).

Язык OWL имеет два основных профиля : OWL full и OWL DL (description logic). OWL DL предназначается для построения онтологий, для которых необходима возможность логического вывода, поэтому на такие онтологии накладываются некоторые ограничения. К таким ограничениям, например относится то, что нельзя описывать класс, который при этом является экземпляром другого класса. OWL Full предназначен для построения онтологий с более мощными синтаксическими возможностями, но при этом без гарантии какого-либо логического вывода. В том, что определенная онтология использует профиль OWL DL, можно убедиться, например с помощью специальной программы-ризонера. Для этого нужно иметь список логических выводов, которые должна делать онтология и воспроизвести данные выводы из этой

онтологии, пользуясь ризонером. Ризонер при этом должен, конечно, поддерживать используемые конструкции RDFS и OWL.

Из сделанного обзора можно сделать следующие **выводы**:

- RDFS и OWL имеют богатую выразительную мощность
- При моделировании ограничений предметной области нужно оставаться в профиле OWL DL, чтобы выполнения моделируемых ограничений можно было проверить.

Глава 2. Обзор тематически близких источников данных и онтологий Linked Open Data

Для того, чтобы понять, как интеграция в LOD выглядит на уже существующих данных, полезно рассмотреть, как именно устроены данные, каким образом применяются термины различных онтологий. Результатом обзора должен стать набор онтологий, пригодных для применения к нашей предметной области, а также набор источников Linked Open Data с указанием способа доступа к ним, которые могут быть использованы для установления внешних связей.

Самым оптимальным решением было бы найти онтологию, полностью моделирующую нашу предметную область, либо найти внешний источник данных, который был бы посвящен в точности теплофизическим свойствам веществ и посмотреть, какими онтологиями пользуется данный источник, а затем провести связывание. Однако в ходе поисков оказалось, что открытых источников по теплофизическим данным в Linked Open Data не опубликовано.

Другим подходом является разбиение предметной области на подобласти и обзор данных и онтологий по этим подобластям. Рассматриваемая предметная область включает несколько предметных областей: вещества, измеряемые свойства и единицы измерения, представление численных значений свойств. Причем представление численных значений свойств не является самостоятельной областью, поэтому соответствующие примеры представления данных лучше искать в наборах посвященным одной из основных самостоятельных областей : вещества и свойства с единицами измерения.

Можно выделить следующую рекомендацию выбора терминов для повторного использования в разрабатываемой онтологии : важно выбрать набор терминов так, чтобы они покрывали большую часть необходимых понятий и связей для публикации наших данных, и в то же время, чтобы использованные термины принадлежали к как можно меньшему числу различных онтологий, то есть чем больше найденная онтология покрывает нашу предметную область, тем лучше.

2.1 Представление химических данных в Linked Open Data

Так как в нашу онтологию необходимо включить тип «Вещество», то нужно найти

подходящий термин в одной из существующих онтологий и использовать его, таким образом, выполняя один из основных принципов [см. 7] Semantic Web. Желательно выбирать онтологии, которые уже широко используются. Поэтому необходимо рассмотреть известные источники химических данных и разобраться в их типизации и связях. Это необходимо также для выбора типов связей на внешние ресурсы, так как они должны быть согласованы с типами этих внешних ресурсов. (Например, неправильно было бы между веществом и молекулой поставить связь owl:sameAs, так как это неверно с точки зрения логики типизации : у нас в наборе рассматриваются именно вещества, а не отдельные молекулы).

Также необходимо исследовать, как устроены данные, представляющие количественные характеристики каких-либо свойств.

При рассмотрении существующих источников химических важно ответить на вопросы :

- Выделяются ли отдельно тип вещества и тип молекулы?
Какие типы используются для описания сущностей веществ или молекул?
Какие предикаты используются для связи вещества и молекул, из которых оно состоит?
- На какие внешние источники химических данных делаются ссылки? Как это влияет на типы описываемых сущностей (на использование типизированной связи в онтологий может быть наложено ограничение, касающееся типов сущностей на концах связи)
- Какие онтологии используются для представления единиц измерения, количественных характеристик и типов свойств веществ? Как выглядит совместное использование этих онтологий?

В итоге, наша цель - выбрать термины для повторного использования в разрабатываемой онтологий.

2.1.1 Краткое описание распространенных химических онтологий

a) Онтология CHEbi (Chemical Entities of Biological Interest);

Приводится классификация химических сущностей, выделяются связи между ними. Есть классы Substance и Pure Substance.

b) Онтология CHEMINF;

Приводится словарь типов химических дескрипторов, например : молекулярная формула, систематическое название. Также есть класс Substance. Определяется предикат, связывающий

вещество и молекулу.

с) Система онтологий ChemAxiom [4];

Разработчики данной онтологии ставят перед собой задачу создать общую онтологию химического домена, которая придет на смену другим, более разрозненным и нецентрализованным онтологиям.

ChemAxiom представляет собой набор онтологий-модулей, каждая из которых описывает отдельную подобласть химии. Все модули являются независимыми, самодостаточными и разрабатываются параллельно. Несмотря на то, что модули являются независимыми, они связаны через онтологию высокого уровня Basic Formal Ontology, а поэтому являются интероперабельными.

Среди модулей, например, есть :

- Главный модуль, где выделяется понятие «Вещество», «Молекула» и связь между ними. Также выделяются классы химических дескрипторов (формула, название и др.);
- Модуль, определяющий свойства веществ (данный модуль представляет собой словарь классов без определенных для них связей с единицами измерения), также в данном модуле вводятся термины для определения состояний вещества;

2.1.2 Источники данных о веществах

а) Набор данных PubChem;

Набор данных PubChemRDF — это RDF версия базы данных PubChem, одного из крупных источников данных в химическом домене.

PubChemRDF не предоставляет SPARQL точку доступа, однако позволяет скачивать RDF данные по FTP. Предполагается, что пользователь, скачав, набор данных или его часть, затем импортирует его в локальное RDF хранилище, к которому уже можно будет обращаться с помощью SPARQL. Еще одним плюсом рассмотрения данного ресурса является то, что PubChemRDF осуществляет интеграцию с Semantic Web на основе онтологий.

Содержимое PubChemRDF включает в себя некоторые связи между химическими соединениями и веществами, хранит для соединений и веществ наборы химических дескрипторов, типизированные связи между соединениями, некоторые метаданные, сведения о

биоактивности некоторых веществ.

Из всего этого набора нас будут интересовать вещества, их дескрипторы (химическая формула, систематическое имя). Также интересным будет способ, которым вещества связываются с соединениями, то есть нас интересуют термины, используемые для этих понятий и для обозначения связи между ними.

PubChem использует ряд стандартизированных онтологий для разных предметных областей. Например, используются следующие онтологии:

- Из химических онтологий используются ChEBI, CHEMINF.
- Онтологии общего назначения: SIO (Semantic Integrated Ontology) , Basic Formal Ontology
- Онтологии единиц измерения : Units of Measurement

Рассмотрим подробнее, каким образом в PubChemRDF используются онтологии CHEMINF, ChEBi. Одним из больших плюсов PubChemRDF является то, что тип сущности «Вещество» отделен от типа сущности «Соединение».

Больше всего нас интересует, как представлено отношение «вещество — состоит_из — соединение», а также типизация сущностей «Вещество» и «Соединение».

На рис. 2 графически изображен фрагмент данных PubChemRDF:

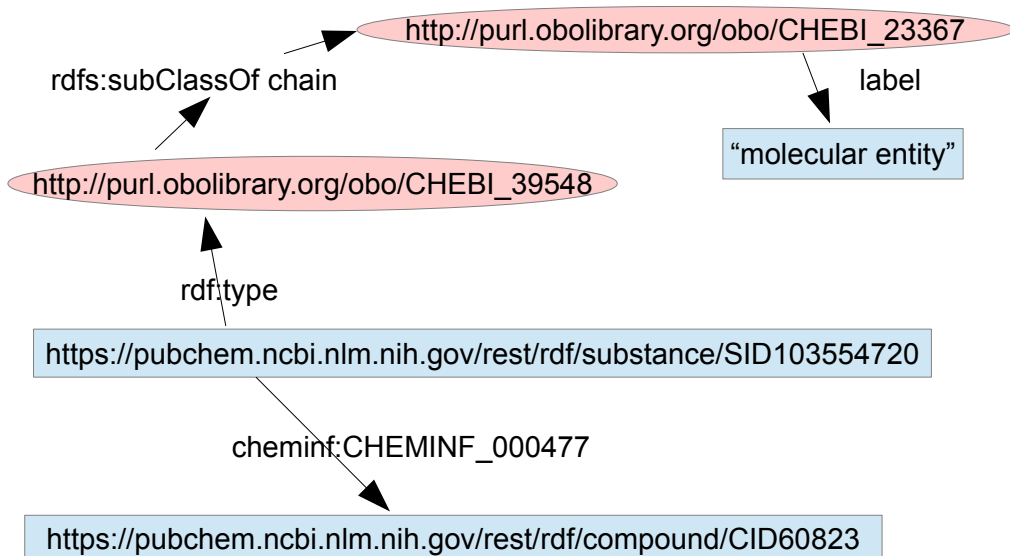


Рис. 2 фрагмент данных набора PubChemRDF

Ресурс-вещество с URI <https://pubchem.ncbi.nlm.nih.gov/rest/rdf/substance/SID103554720> ссылается на ресурс-соединение с URI <https://pubchem.ncbi.nlm.nih.gov/rest/rdf/compound/CID60823> с помощью типизированной связи cheminf:CHEMINF_000477, определенной в онтологии CHEMINF: «has PubChem normalized counterpart». Данный предикат является специфичным и может использоваться только для привязки нашего набора данных с PubChemRDF.

Для поиска подходящего предиката рассмотрим определение cheminf:CHEMINF_000477, данное в онтологии CHEMINF, а также определения свойств, стоящих выше по иерархии наследования.

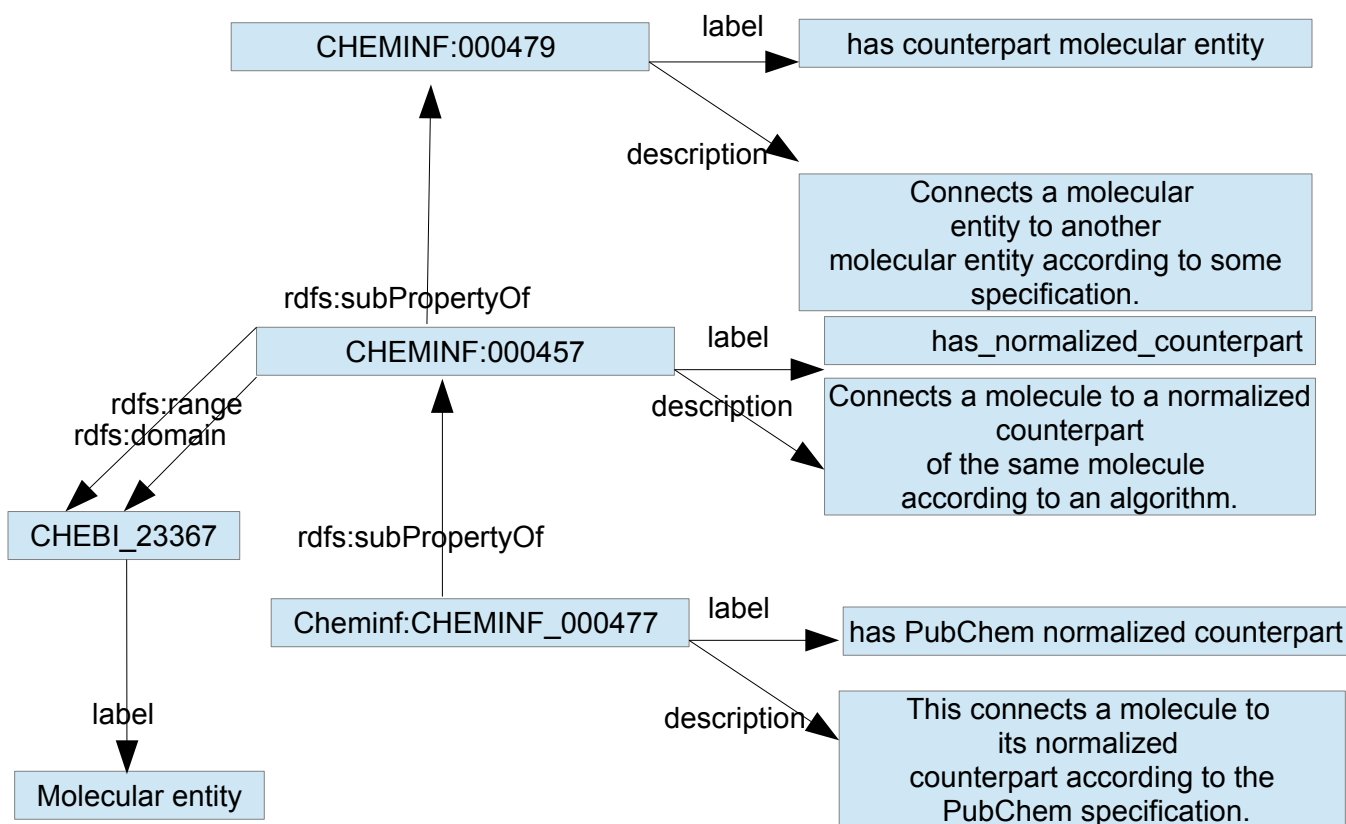


Рис.3 Иерархия свойств, определяющих отношения сущностей-веществ и сущностей-молекул в онтологии CHEMINF

Как видно из рис. 3, для свойства `has_normalized_counterpart` (CHEMINF:000457) указаны ограничения на типы сущностей (`rdfs:domain` и `rdfs:range`): сущности на концах связи должны принадлежать классу `Molecular Entity` (http://purl.obolibrary.org/obo/CHEBI_23367) из онтологии

CHEbi.

Как указано в описании (description) данного класса в онтологии, он должен использоваться для типизации любых отдельных атомов, молекул, ионов, пар ионов, и т. д., которые можно идентифицировать как отдельные сущности. Очевидно, понятие «Вещество» под такое определение не подходит.

На рис. 4 графически показана часть определения класса Molecular Entity.

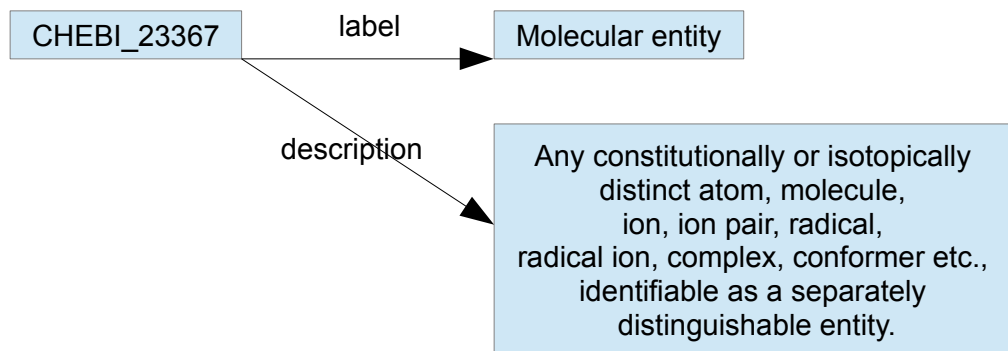


Рис. 4 Фрагмент определения класса Molecular Entity онтологии CHEMINF

Исходя из данного определения и приведенного примера данных можно сделать следующие **выводы** :

- В PubChem разделение на «Вещества» и «Соединения» проходит только на уровне данных и является условным, формально выводимый тип для обоих видов сущностей - «Molecular entity» из онтологии CHEbi;
- Классы для типизации веществ из онтологий CHEbi и CHEMINF не используются;
- Для связи «Вещество → состоит из → Молекула» используется свойство «has PubChem normalized counterpart», которое не помогает в разделении понятий «Вещество» и «Молекула»;

Как ресурсы-вещества, так и ресурсы-соединения в PubChemRDF имеют множество атрибутов, как правило это внутренние или общепринятые идентификаторы или же атрибуты, определяемые для отдельной молекулы вещества : молекулярная формула, полный заряд, молярная масса и т.д. Все виды подобных идентификаторов и атрибутов определены в онтологии CHEMINF. Для каждого вещества(соединения) для каждого его атрибута заводится

отдельная сущность-атрибут, тип которой определен в CHEMINF.

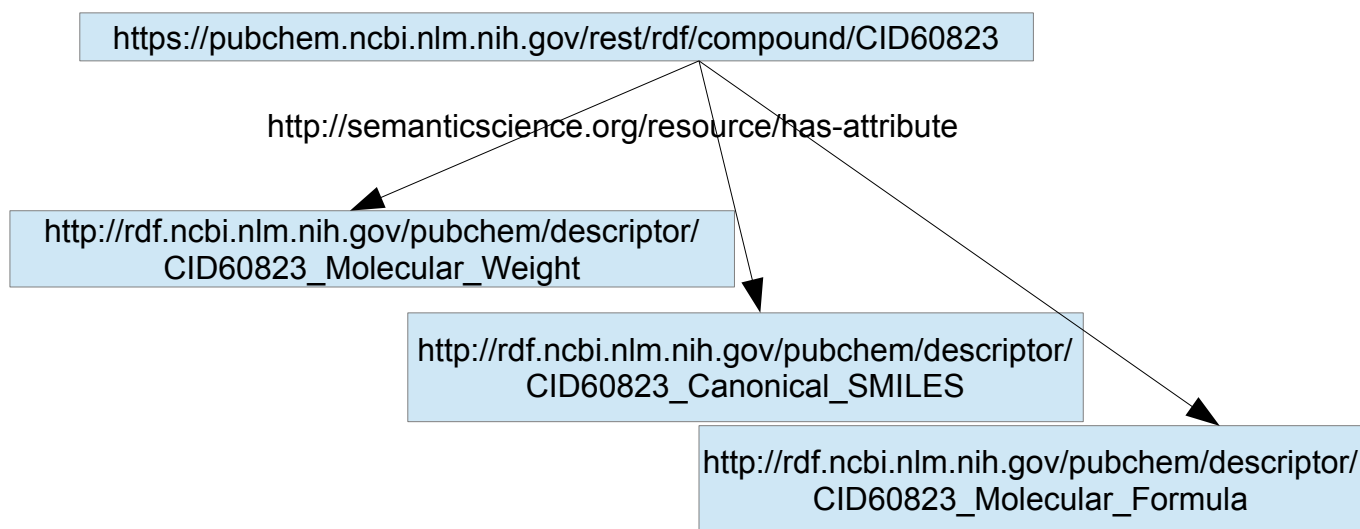


Рис. 5 Фрагмент данных PubChem : ресурс-химическое соединение с ссылками на атрибуты

Из рис. 5 видно, что сущность-химическое соединение ссылается на сущность-атрибут с помощью предиката `sio:has_attribute` из онтологии Semantic Integrated Ontology.

Среди таких атрибутов нам могут пригодиться : молекулярная формула (`http://semanticscience.org/resource/CHEMINF_000335`), систематическое имя, а также все атрибуты, связанные с упоминанием систематического имени (`http://semanticscience.org/resource/CHEMINF_000382`).

Для описания атрибутов, которые предполагают количественное значение и указание единиц измерения используются термины онтологий Sio (`http://semanticscience.org/resource/has-value`, `http://semanticscience.org/resource/has-unit`) и Units of Measurement (отсюда используется словарь единиц измерения).

Предикат `sio:has_unit` из онтологии Sio накладывает ограничение на класс субъекта отношения : в онтологии Sio определен класс «Unit of measurent» (единица измерения), и субъект отношения `has_unit` обязан принадлежать к данному классу. В наборе PubChem в качестве представления единиц измерения используется онтология Units Of Measurement, где единицы измерения даются в виде словаря классов. Таким образом мы видим, что в PubChem предикат `has_unit` ссылается на объекты типа `owl:Class`, что выходит за рамки OWL DL.

PubChem связан с другим набором данных химического домена— ChEMBL. Ссылки на соответствующие ресурсы в ChEMBL RDF имеют тип `skos:closeMatch`. Данный тип связи не накладывает никаких ограничений на типы сущностей на концах связей.

b) Набор данных ChEMBL [11];

Одним из ресурсов, предоставляющих данные о химических веществах является ChEMBL. ChEMBL содержит описания биоактивных молекул. Предметная область данного ресурса относится скорее к биологии, чем к химии, однако ChEMBL содержит значительное количество записей о веществах и их основных параметрах, а также ссылки на внешние источники, поэтому рассмотреть данный ресурс было бы полезно для восстановления полной картины взаимосвязей в химическом LOD, а также чтобы посмотреть как типизируются химические сущности.

Данная база выложена в виде Linked Open Data в двух независимых версиях :

- RDF Platform>ChEMBL (<http://www.ebi.ac.uk/rdf/>);
- ChEMBL-RDF (<http://datahub.io/dataset/farmbio-chembl>);

Первый источник использует собственную внутреннюю онтологию для описания основных классов и связей модели данных. Кроме того Sparql точка RDF Platform>ChEMBL имеет ограничения по размеру запрашиваемых данных, и поэтому не совсем удобная для связывания. Данный источник пока не имеет ссылок на внешние ресурсы Linked Open Data.

Второй источник аналогично примерам PubChem используют онтологии ChEMINF и ChEBi, главным типом сущности является молекула. Числовых значений свойств в наборе не присутствует, есть только идентификаторы(формула, название и др.) ChEMBL-RDF определяет множество внешних связей, например с ресурсом OpenMolecules RDF, который в свою очередь связан с ChemSpider. На ресурсы этого источника проставлено множество ссылок из PubChem, рассмотренного выше. Доступ к ChEMBL-RDF предоставлен с помощью SPARQL точки.

c) Набор данных ChemSpider;

ChemSpider представляет из себя базу веществ и их свойств, в том числе и теплофизических. Для каждого вещества можно получить RDF-документ с информацией об этом веществе, однако на данный момент данных по свойствам веществ в получаемых RDF-документах нет. Присутствуют только общие данные, характеризующие вещество (название, формула, различные идентификаторы).

Главным типом сущности является «Вещество», для описания данных используется онтология ChemAxiom.

SPARQL-точки у этого набора пока нет, однако функционирует точка поиска, работающая по протоколу HTTP, которой можно посылать запросы поиска по веществам, в качестве ответа выдается либо RDF-документ, если под критерии поиска подходит только одно вещество, либо XML-данные со списком параметров найденных веществ, если таких веществ несколько.

В результате обзора химических данных в Linked Open Data можно сделать следующие выводы :

- В большинстве источников данных, посвященных веществам и химии, не делается четкое различие между понятием молекулы и понятием вещества;
- Не было обнаружено примеров использования классов из онтологий CHEMINF и Chebi, обозначающих понятие «Вещество»;
- Для представления данных о числовых значениях свойств используются термины Semantic Integration Ontology и Units of Measurement способом, выходящим за рамки OWL DL;
- Среди рассмотренных источников только в ChemSpider главный типом сущностей является «Вещество»;
- ChemSpider также является единственным источником, специализирующимся именно на физико-химических свойствах веществ, а не на биологической тематике, как другие наборы (хотя данные о свойствах и отсутствуют в самих RDF-данных);
- Рассмотренные наборы данных взаимосвязаны;

2.2 Измеряемые свойства и единицы измерения в Linked Open Data

Рассмотрим предметную область свойств и единиц измерения.

В Linked Open Data не нашлось источника, предоставляющего данные о свойствах и единицах измерения, однако существуют онтологии, которые определяют словари свойств и единиц измерения.

a) Semantic Integration Ontology;

При рассмотрении данных химических наборов, нам уже встречалась онтология Semantic

Integration Ontology, в наборе PubChem из нее использовались предикаты `has_attribute`, `has_unit`, `has_value`. Данная онтология служит для представления данных о научных исследованиях. Данная онтология включает классы `Quantity` (величина) и `Unit of measurement` (единица измерения), для указания числового значения используется предикат `has_value`, для указания единицы измерения используется предикат `has_unit`. Класс `Quantity` имеет набор подклассов-величин, но они подходят скорее для представления химических данных : задается ограниченный набор таких свойств как : концентрация, количество вещества, а также некоторые разновидности пространственных свойств (длина, объем). Свойства в данной онтологии определены как классы, а для единиц измерения соответствующего словаря нет.

b) ChemAxiomProp онтологии ChemAxiom;

Онтология `ChemAxiom` уже упоминалась выше при рассмотрении области `Linked Open Data`, посвященной химии и веществам. В данной онтологии присутствует модуль, задающий список свойств веществ. Свойства представлены в виде классов. Единиц измерения для данных свойств не представлено. Данная онтология находится в разработке.

c) Онтология единиц измерения The Ontology of Units of Measurement;

Данная онтология задает словарь единиц измерения, разделяя их на классы. При этом словаря свойств, которые могут быть измерены указанными единицами не приводится.

d) Онтология QUDT (Quantities, Units, Dimensions and Data Types);

Это набор онтологий, который определяет основные классы для свойств, из размерностей, единиц измерения, величин и их числовых значений. Вводятся словари свойств и единиц измерения.

Основные классы онтологии `QUDT` изображены на рис. 6.

`Quantity` (величина) это наблюдаемое свойство объекта или системы, которое может быть измерено и которому можно поставить в соответствие численное значение. Для `Quantity` указывается два основных параметра : вид измеряемого свойства (предикат `quantityKind`) и численное значение величины (предикат `quantityValue`). `QuantityKind` (вид измеряемого свойства) это вид свойства, которое можно измерить, например длина, сила и т.д.

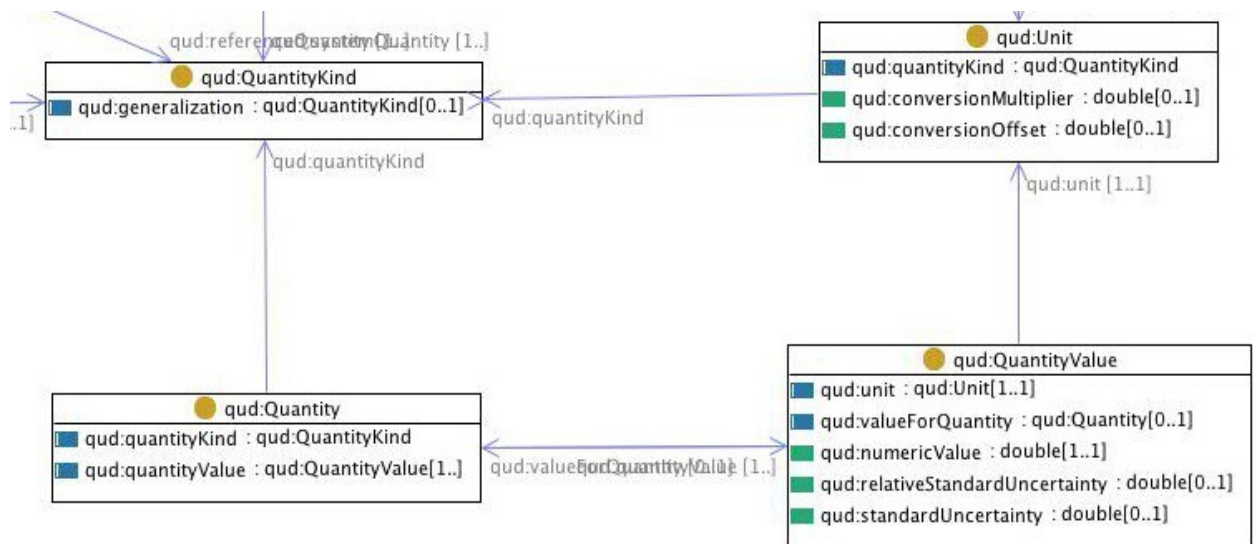


Рис. 6: Основные классы онтологии QUDT

QuantityValue (численное значение величины) выражает численную характеристику заданной величины, при этом указывается единица измерения, для которой записано это численное значение. В качестве основных параметров имеет соответствующую величину (предикат valueQuantity) и непосредственно численное значение (предикат numericValue)

Unit (единица измерения) — это определенная численная величина заданного типа свойства, которую выбрали как единичное численное значение данного свойства, относительно которого пропорционально будут указываться все численные значения данного свойства. Параметры единицы измерения это вид измеряемого свойства (предикат quantityKind), а также параметры преобразования в базовую единицу измерения (предикаты conversionMultiplier и conversionOffset). Дело в том, что измеряемое свойство (QuantityKind) может измеряться несколькими единицами; и, чтобы можно было выразить численное значение величины в одной единицы измерения через другую единицу, были введены указанные параметры преобразования.

Для того, чтобы совершить преобразование численного значения value из единицы измерения unit1 в другую- unit2, нужно применить следующую формулу :

$$((value + unit1.offset) \cdot unit1.multiplier - unit2.offset) \div unit2.multiplier$$
, где unit1.offset и unit2.offset – параметры смещения, указанные для данных единиц измерения, а unit1.multiplier и unit2.multiplier – коэффициенты пропорциональности относительно некоей базовой единицы измерения. Естественно, данное преобразование будет иметь смысл, только если указанные единицы измерения сравнимы.

Словарь свойств и единиц измерения представляет из себя список экземпляров классов QuantityKind и Unit, в этом смысле онтология QUDT можно рассматривать как набор данных.

Для типов измеряемых величин (QuantityKind) в QUDT выделено множество подклассов, отображающих принадлежность данного типа свойства к определенной области (рис. 7). Для единиц измерения также выделены подклассы, соответствующие областям применения тех или иных единиц (рис. 8).

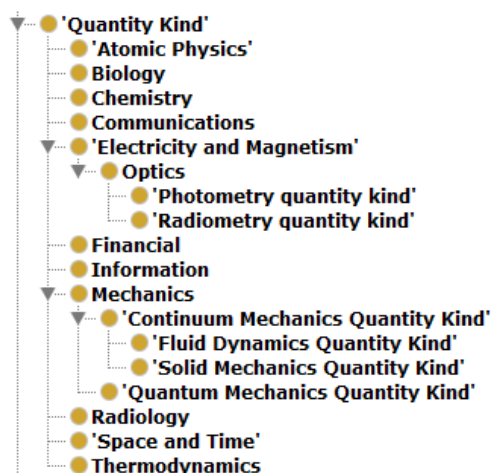


Рис. 7: Категории типов свойств QUDT

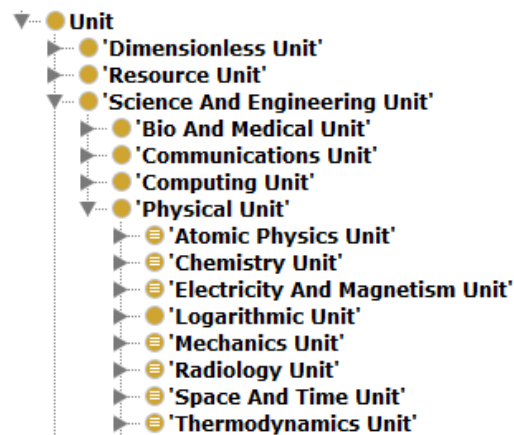


Рис. 8: Категории типов единиц измерения QUDT

Одним из важнейших предикатов онтологии QUDT является предикат `qudt:generalization`, он ставится между двумя экземплярами класса `QuantityKind` и обозначает, что одно свойство является обозначением другого, оба указанных свойства имеют одинаковую размерность и к ним применимы одни и те же единицы измерения.

Например, нам нужно завести в словаре свойств кинетическую энергию и потенциальную энергию, но мы знаем, что оба это свойства — это частные виды энергии, а для типа свойства «Энергия» у нас уже есть единицы измерения. Таким образом, мы укажем, что обобщением (`generalization`) типа свойства «Кинетическая энергия» и типа свойства «Потенциальная энергия» является тип свойства «Энергия». Единицы измерения являются сравнимыми, если те типы свойств, которые они измеряют, имеют одинаковое обобщение (`qudt:generalization`).

В онтологии QUDT также выделяется ряд классов для работы с размерностями. Смысл заключается в том, что в разных системах величин базовыми являются определенные типы свойств, а другие выводятся из них, таким образом размерности для одного и того же типа свойства в разных системах величин могут быть разными. QUDT предоставляет возможности для записи размерностей типов свойств в разных системах единиц, а также вводит целый

словарь размерностей для разных типов свойств(это также экземпляры классов). При этом полученными данными можно пользоваться чтобы, например проверять совпадение размерностей левой и правой стороны алгебраического выражения, в котором участвуют типы свойств.

Из обзора онтологий по единицам измерения и измеряемым свойствам можно сделать **вывод**: из рассмотренных онтологий только QUDT предоставляет и классы для представления величин и их значений, а также предикаты для связи экземпляров этих классов, и в то же время списки типов свойств и единиц измерения. Данная онтология покрывает заданную область в наибольшей степени.

2.3 Общие выводы из обзора тематически схожих источников данных и онтологий

- Точно совпадающих по тематике наборов данных или онтологий обнаружено не было;
- В ходе обзора были рассмотрены источники данных и онтологии, посвященные веществам и химии;
 - Из рассмотренных источников химических только ChemSpider совпадает с нашими данными по предмету рассмотрения (главная сущность - «Вещество», рассматриваются физико-химические свойства);
 - Из рассмотренных химических онтологий только для ChemAxiom найдены примеры применения класса, обозначающего понятие «Вещество». Кроме того данная онтология является более новой разработкой по сравнению с другими рассмотренными онтологиями. При этом данная онтология используется для представления данных источника ChemSpider, подходящего для связывания;
- Также были рассмотрены онтологии, определяющие классы и связи для представления свойств, их численных значений и единиц измерения;
 - Среди таких онтологий онтология QUDT оказалась самой богатой в плане предоставляемых терминов. Она определяет классы и для единиц измерения, и для свойств, и для величин и их значений, определяет словари типов свойств и единиц измерения, в то время, как другие рассмотренные онтологии определяют средства представления лишь для частей данной предметной области. Возможности работы с размерностями не предоставляет ни одна из рассмотренных онтологий, кроме QUDT;

- Онтология QUDT определяет типы свойств и единиц измерения как экземпляры классов, поэтому можно рассматривать ее как внешний источник-кандидат для связывания;

Глава 3. Подход к интеграции данных по свойствам веществ в пространство Linked Open Data.

3.1 Построение онтологии предметной области

Основные классы и связи для требуемой онтологии были определены в работе [2]. Также в результате упомянутой работы была разработана схема реляционной базы данных для работы с данными по свойствам веществ. Далее будем отталкиваться от данной схемы, так как она более полно и конкретно отражает специфику исходных данных, при этом придерживаясь структуры, введенной в [2]. Графическое изображение указанной схемы дано в приложении А.

3.1.1 Повторное использование терминов из существующих онтологий

В результате обзора области Linked Open Data и онтологий с тематикой, касающейся веществ, а также измеряемых свойств и их численных значений, были сделаны выводы, которые определяют выбор онтологий, которые выгоднее всего использовать для решения поставленной задачи.

Однако следует помнить, что исходные данные представлены в виде реляционной базы, и их придется преобразовать в RDF формат, согласованный с разрабатываемой онтологией. Поэтому для того, чтобы составить требуемую онтологию нужно, пользуясь схемой

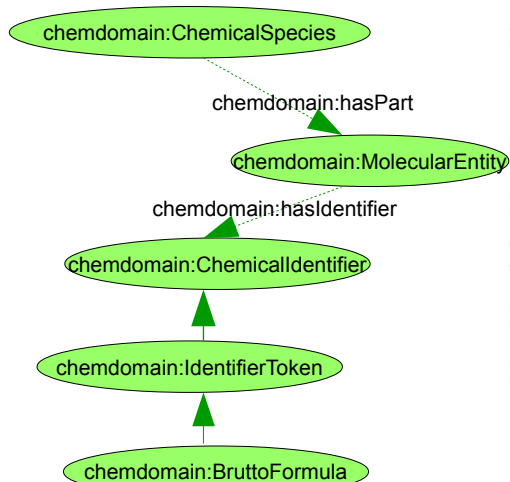


Рис. 9: Классы для представления данных о веществе

предоставленной реляционной базы, построить соответствие таблицам данной реляционной базы классов из предметной области. Также необходимо выбрать инструмент преобразования данных реляционной базы в RDF формат и убедиться, что исходные данные при использовании выбранного инструмента могут быть представлены в виде, соответствующем получившейся онтологии. От выбора инструмента преобразования также будет зависеть способ хранения связей данных из нашего набора с внешними источниками.

1. Рассмотрим онтологию ChemAxiom и часть схемы исходной реляционной базы, которая посвящена информации о веществах. На рис. 3 графически показаны классы, определенные для представления сущности типа «Вещество» и ее идентификаторов. На рис. 3 показано, что экземпляр класса ChemicalSpecies (вещество) может ссылаться на экземпляр класса MolecularEntity с помощью предиката has:part, а экземпляр класса MolecularEntity может ссылаться на экземпляр класса ChemicalIdentifier с помощью предиката hasIdentifier. В ChemAxiom дается классификация типов идентификаторов, но нам понадобится только формула (BruttoFormula), и, возможно, систематическое название.

Таблица substance исходной базы хранит для каждого вещества имя и формулу. Таким

public.substance

id: bigint
version: bigint
name: varchar(255)
subst_formula: varchar(255)

образом мы должны каждую запись в таблице substance преобразовывать в три RDF ресурса классов ChemicalSpecies, MolecularEntity и BruttoFormula, связанных соответствующими предикатами.

Рис. 10: Таблица, хранящая данные по веществам из исходной базы

2. Рассмотрим часть схемы исходной реляционной базы, посвященную информации о единицах измерения, свойствах и их численных значениях, и онтологию QUDT.

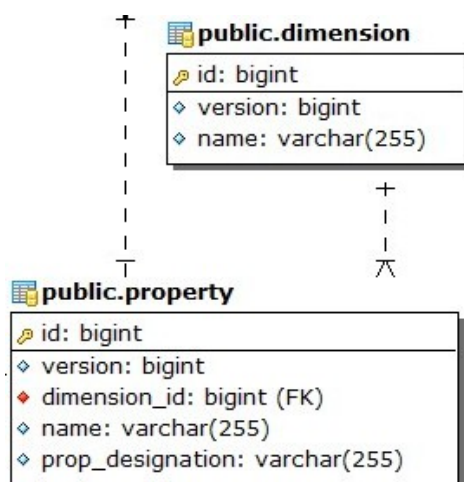
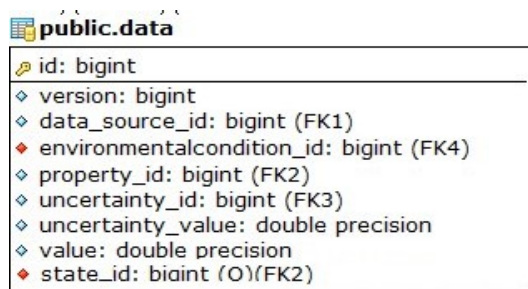


Рис. 10: Таблицы для хранения информации о свойствах и единицах измерения

Таблицы property и dimension хранят информацию об измеряемых свойствах и их единицах измерения. Как видно из схемы, они связаны отношением «один ко многим». В онтологии QUDT сущности сходных классов QuantityKind и Unit связаны отношением «многие ко многим», кроме того в онтологии QUDT единица измерения имеет параметр-ссылку на измеряемое свойство, а не наоборот, как у нас в схеме. Таким образом при использовании классов QuantityKind и Unit нам придется менять направление связи «свойство — единица измерения» в обратную сторону. Важно, чтобы инструмент преобразования из реляционной базы в RDF формат позволил поменять направление связи таким

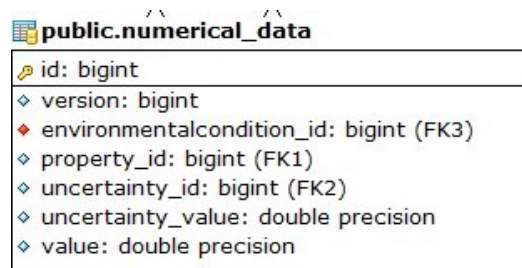
образом. В нашей базе также не предусмотрено хранение дополнительной информации о

свойствах и единицах измерения, которая предусмотрена в QUDT.



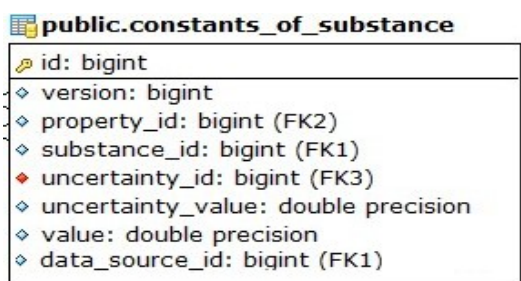
id: bigint
version: bigint
data_source_id: bigint (FK1)
environmentalcondition_id: bigint (FK4)
property_id: bigint (FK2)
uncertainty_id: bigint (FK3)
uncertainty_value: double precision
value: double precision
state_id: bigint (O)(FK2)

Рис. 11: Таблица, хранящая численные значения свойств-функций



id: bigint
version: bigint
environmentalcondition_id: bigint (FK3)
property_id: bigint (FK1)
uncertainty_id: bigint (FK2)
uncertainty_value: double precision
value: double precision

Рис. 12: Таблица, хранящая численные значения, характеризующие условия среды



id: bigint
version: bigint
property_id: bigint (FK2)
substance_id: bigint (FK1)
uncertainty_id: bigint (FK3)
uncertainty_value: double precision
value: double precision
data_source_id: bigint (FK1)

Рис. 13: Таблица, хранящая численные значения свойств-констант

Таблицы `numerical_data`, `constants_of_substance` и `data` рассмотрим вместе, так как они представляют схожие типы сущностей для предметной области. Структура этих таблиц изображена на рис. 11,12,13. Таблица `data` хранит информацию о численных значениях свойств веществ в определенных условиях среды. Таблица `numerical_data` хранит информацию о численных значениях каждого из параметров условий среды, от которых зависят рассматриваемые свойства веществ и их значения. Связь таблиц `data` и `numerical_data` поясним позже. Таблица `constants_of_substance` хранит информацию о численных значениях константных свойств веществ, у которых нет аргументов.

У рассмотренных таблиц есть общий набор полей :

- `property_id` – ссылка на рассматриваемое свойство;
- `value` – численное значение рассматриваемого свойства;
- `uncertainty_id` — ссылка на запись, хранящую информацию о типе погрешности для значения `value`;

- `uncertainty_value` – численное значение погрешности;

Для `data` и `constants_of_substance` общим является поле `data_source_id` — ссылка на источник информации о числовом значении, указанном в `value`.

Вспомним, что в онтологии QUDT отдельно определяется класс `Quantity` (величина), который должен хранить информацию об условиях и объекте измерения, в том числе в QUDT введен предикат `qudt:quantityKind` (ссылка на измеряемое свойство); отдельно определяется класс `qudt:QuantityValue` (значение величины), который хранит информацию о конкретном численном значении величины.

Исходя из описанного можно сделать вывод, что сущности, информация о которых хранится в указанных таблицах исходной базы, можно разбить на пары экземпляров классов `qudt:Quantity` (величина) и `qudt:QuantityValue`, связанные предикатом (`qudt:quantityValue`). В онтологии QUDT `Quantity` и `QuantityValue` находятся в отношении «один ко многим» (значение величины может быть выражено в различных единицах измерения), у нас же получается отношение «один к одному», что однако не мешает представить данные в требуемом виде.

Сущность `Quantity` будет обладать ссылкой (предикат `qudt:quantityKind`) на измеряемое свойство (ссылка `property_id`), а также ссылкой на сущность типа `QuantityValue`, у которой будет параметр- численное значение (предикат `qudt:numericValue`). Однако здесь не хватает обязательной ссылки на единицу измерения (предикат `qudt:unit`), но мы можем вычислить значение недостающей ссылки, сделав несложный запрос, который выясняет, с помощью какой единицы измерения, может быть измерено свойство, значение для которого рассматривается (в исходной базе у каждой записи в таблице `property` есть ссылка на запись в таблице `dimension`). *Здесь важно, чтобы инструмент преобразования из реляционной базы в RDF предоставлял возможность делать такой запрос при преобразовании данных.*

Для обозначения параметров, не предусмотренных QUDT можно ввести свои собственные предикаты. В том числе можно ввести класс `QuantityValueFromSource`, обозначающий значение величины, взятое из литературы. Для связи экземпляра класса `QuantityValueFromSource` и экземпляра класса `DataSource` введем предикат `dataSource`. Введем также классы `Data`, `SubstanceConstant` и `NumericalData`, унаследовав их от класса `qudt:Quantity`. Определим дополнительные параметры, характеризующие условия измерения рассматриваемой величины, которые не предусмотрены в QUDT и являются специфичными для наших данных. Например,

определим новый предикат `substance`, который будет соединять константу вещества с самим веществом.

Определим класс `UncertaintyType`, экземплярами которого будут типы погрешностей измеряемых величин. Так как набор классов погрешностей не определен заранее, а является открытым списком, мы не можем включить в онтологию перечисление типов погрешностей в виде классов. Заведем специальный предикат `uncertaintyType` для ссылки на экземпляры класса `UncertaintyType`, а также предикат `uncertaintyValue` для указания численного значения погрешности указанного класса.

На данный момент схему моделируемой онтологии можно представить следующим образом:

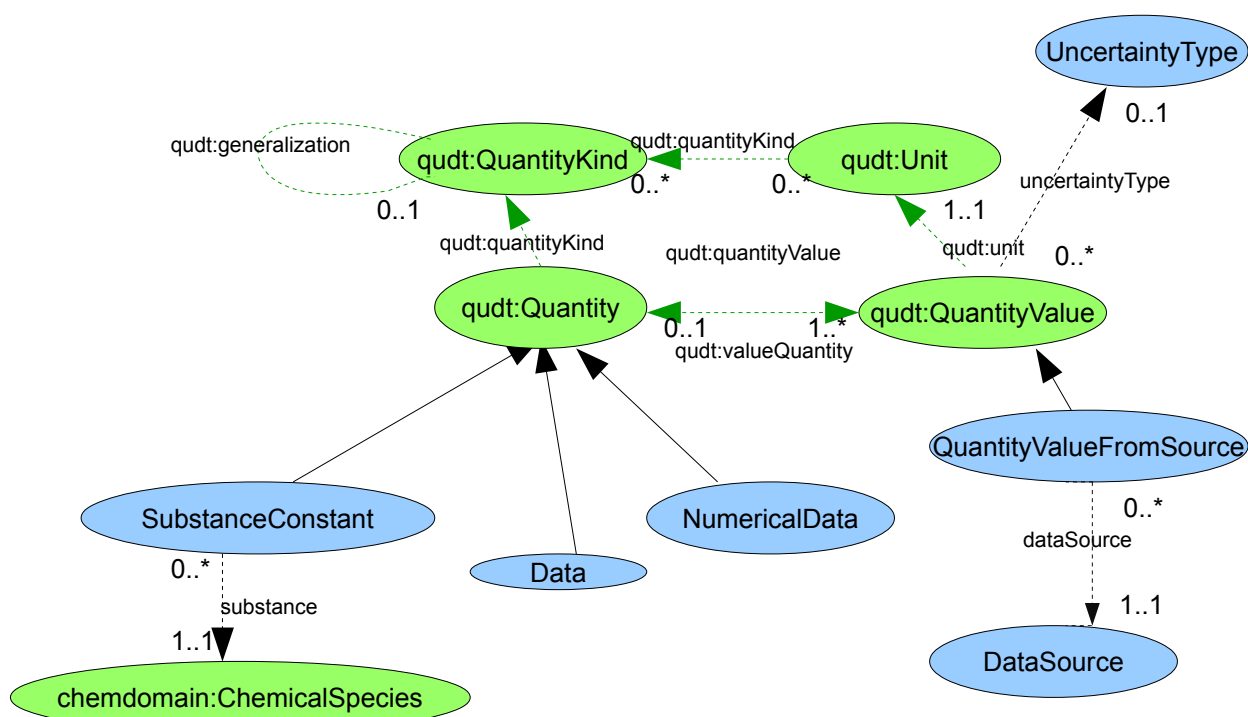


Рис. 14: Схема моделируемой онтологии (*DatatypeProperty* не показаны)

Отметим, что для классов `SubstanceConstant` и `Data` нужно ввести ограничение на тип объекта онтошения задаваемого предикатом `qudt:quantityValue`, это обязательно должен быть объект из класса `QuantityValueFromSource`. То есть величины констант веществ и неконстантных свойств должны иметь в качестве числового значения значение, взятое из литературы.

3. Рассмотрим подробнее, каким образом в исходной схеме задается соответствие значений свойств функций от значений свойств-аргументов.

В исходной схеме предусмотрена вспомогательная таблица `environmental_conditions`.

 **public.environmental_condition**

id: bigint
version: bigint
state_id: bigint (FK2)
substance_id: bigint (FK1)
text_description: varchar(255)

Рис. 15: Таблица, связывающая набор значений свойств вещества с условиями среды

Через эту таблицу организовывается связь значений свойств-функций со значениями свойств-аргументов. Записи таблицы представляют набор условий среды, то есть набор значений свойств-аргументов. Таблицы `data` и `numerical_data` имеют общее свойство-ссылку на таблицу `environmental_conditions`. В случае `data` эта ссылка обозначает условия среды, в которых рассматривается значение

свойства-функции, а в случае `numerical_data` — принадлежность к набору значений функций-аргументов (набору условий среды). Таким образом, сетка температур, например, может быть задана набором записей в таблице `environmental_conditions` (одна запись, для одного значения температуры), а для каждой записи в `environmental_conditions` — одной записью в `numerical_data` (конкретное значение температуры).

Отметим также, что каждая запись в `environmental_conditions` делается для определенного вещества (`substance_id`) в определенном фазовом состоянии (`state_id`). То есть если нужно для различных веществ рассмотреть значения свойств на сетке температур, то необходимо завести такую сетку (набор записей в `environmental_conditions` и `numerical_data`).

Hydroxyl OH(g)

$\Delta_f H_0^\circ$	39.110 kJ/mol	$\Delta_f H_{298}^\circ$	39.349 kJ/mol
----------------------	---------------	--------------------------	---------------

T	Cp	F	S	H
K	J/K*mol	J/K*mol	J/K*mol	kJ/mol
298.15	29.886	154.068	183.627	8.813
300.00	29.879	154.251	183.812	8.868
400.00	29.606	162.763	192.366	11.841
500.00	29.487	169.368	198.957	14.795
600.00	29.512	174.761	204.333	17.743
700.00	29.663	179.319	208.892	20.701
800.00	29.921	183.270	212.869	23.679
900.00	30.265	186.759	216.412	26.688
1000.00	30.674	189.887	219.622	29.735
1100.00	31.126	192.726	222.566	32.824
1200.00	31.601	195.328	225.295	35.961
1300.00	32.077	197.732	227.843	39.145
1400.00	32.533	199.969	230.237	42.375
1500.00	32.947	202.063	232.496	45.650
1500.00	32.904	202.063	232.494	45.646
1600.00	33.370	204.032	234.632	48.960
1700.00	33.781	205.893	236.668	52.318
1800.00	34.149	207.657	238.609	55.715

Рис. 16: Фрагмент исходных данных

соответствующего значения температуры вещества Hydroxyl в состоянии газа.

Здесь мы подходим к центральной сущности в данной предметной области — **набору данных о веществе**. (Важно не спутать этот термин с термином «набор данных в Linked Open Data».) Под набором данных о веществе понимается набор значений различных свойств-функций в заданных условиях среды для заданного вещества в заданном фазовом состоянии.

Приведем еще раз для наглядности иллюстрацию из введения.

В данном случае каждая строка таблицы — это набор данных для

Таким образом, из каждой записи таблицы `environmental_conditions` логично было бы выделить сущности двух типов : «условия среды» (`EnvironmentalConditions`) и «Набор данных о веществе» `Dataset`. В общем случае эти экземпляры этих классов должны быть связаны как «один ко многим», в исходной базе эти сущности связаны как «один к одному», потому что изначально представляют собой одну запись в таблице `environmental_conditions`.

На рис. 17 представлен фрагмент разрабатываемой онтологии, в котором указаны классы `Dataset` и `EnvironmentalConditions` и их связи с другими уже рассмотренными классами.

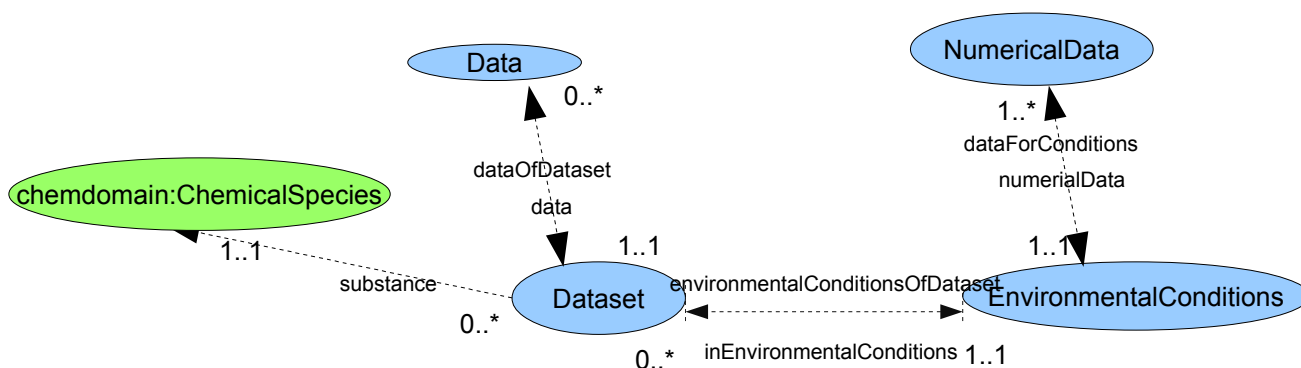


Рис.17: Фрагмент разрабатываемой онтологии: `Dataset` и `EnvironmentalConditions`

Полное графическое представление онтологии дано в приложении В.

Итог :

- На основе результатов [2] была разработана структура онтологии для представления данных о свойствах веществ, были использованы некоторые классы и предикаты из других онтологий (QUDT, ChemAxiom), а для тех типов сущностей и связей;
- Были определены виды преобразования, которые могут потребоваться для представления исходных данных в виде, соответствующем полученной онтологии. Эти преобразования необходимы потому что, как видно из построения, разработанная онтология более подробно описывает предметную область, исходная онтология [2] и, соответственно, чем схема исходной базы данных.

3.1.2 Анализ ограничений предметной области

В соответствии с поставленной задачей нужно проанализировать ограничения, которые накладывает рассматриваемая предметная область, и выяснить, каким образом их можно промоделировать с помощью языка OWL.

1. Простейшие ограничения на domain и range введенных нами предикатов.

Напомним, что owl:domain и owl:range задают ограничения на типы экземпляров на концах связи, в которой присутствует рассматриваемый предикат.

Практически все введенные нами новые предикаты используются только для связи двух классов (кроме предиката substance, который используется для указания вещества для экземпляров Dataset, а также для указания вещества для экземпляров SubstanceConstant).

Поэтому для подавляющего большинства предикатов мы можем жестко зафиксировать классы, к которым принадлежат экземпляры, участвующие в тройках вместе с этими предикатами.

Например для предиката data, который служит для перечисления данных в наборе можно записать :

```
data rdfs:type ObjectProperty.
```

```
data rdfs:domain Dataset.
```

```
data rdfs:range Data.
```

Для того, чтобы не писать аналогичную конструкцию для обратного предиката dataOfDataset, объявим этот предикат инверсным по отношению к рассмотренному data:

```
dataOfDataset rdf:type ObjectProperty.
```

```
dataOfDataset owl:inverseOf data.
```

Аналогично зададим domain и range для всех остальных предикатов, которые используются для связи лишь одной пары классов. Для единственного предиката uncertaintyValue типа rdfs:DatatypeProperty зададим в качестве range тип double.

2. Ограничения, задаваемые с помощью owl:cardinality

Самым простым видом ограничений являются ограничения на количество триплетов с данным предикатом для экземпляра заданного класса. Такие ограничения задаются с помощью конструкции Restriction с указанным значением для предикатов owl:minCardinality и/или

owl:maxCardinality.

Например, зададим, ограничение, которое говорит, что для экземпляра класса Data должен быть указан экземпляр класса Dataset:

```
Data rdfs:subClassOf Restriction [  
  owl:onProperty dataset.  
  owl:minCardinality 1.  
]
```

Аналогично определим ограничения и для других связей в онтологии.

3. Более сложные ограничения предметной области.

а) Свойство-функция / свойство-константа:

Предполагается, что свойства могут быть функциями(численные значения зависят от условий среды) и могут быть константными(численные значения не зависят от условий среды). Эти два класса не пересекаются;

б) Аргументы свойства-функции:

У свойств-функций должны быть определены аргументы (некоторые другие измеряемые свойства);

Определения условий среды для каждого из фрагментов данных, содержащих численные значения рассматриваемых свойств-функций должны использовать только указанные свойства-аргументы;

с) Характер монотонности свойства-функции:

У свойств-функций может быть разный характер монотонности, с которым должны быть согласованы хранимые численные значения этих свойств;

д) Область определения свойства-функции:

Для свойства-функции как правило задана область определения, с которой должны быть согласованы с численные характеристики условий среды, заданных для значений рассматриваемого свойства;

е) Множество значений свойства-функции:

Для свойства-функции как правило задано множество значений. Все численные значения данного свойства-функции должны принадлежать указанному множеству значений.

Множества значений и определения для свойства-функции как правило является

отрезком. Например [0, 1000].

Границы указанных множеств могут быть выражены в разных единицах измерения, которые применимы к данному свойству.

- f) Согласованность ссылок на состояния вещества при определении наборов данных, фрагментов данных и применимости свойств-функций к состояниям:

Свойства могут быть применимы к одним фазовым состояниям вещества и неприменимы к другим. Набор данных для вещества (как уже описано выше, это - набор значений свойств вещества в определенных условиях среды и в определенном фазовом состоянии) вводится для определенного фазового состояния вещества.

Если фазовое состояние набора включает в себя более одного агрегатного (например, «газ-жидкость»), то

1. для тех рассматриваемых свойств-функций, которые применимы по-отдельности к агрегатным состояниям (например, «газ», «жидкость») для численного значения должно уточняться, для какое конкретно из агрегатных состояний оно задано.
2. Бывают свойства, о которых имеет смысл говорить лишь в контексте «смешанного состояния» (например «газ-жидкость»), для их численных значений уточнений о состоянии не делается.

4. Анализ ограничений и способы из моделирования.

Для того, чтобы была возможность проверить какие-либо ограничения, необходимо

- a) иметь данные, которые надо проверять на соответствие ограничениям,
- b) иметь отдельно записанные ограничения.

Для простых ограничений из пунктов 1 и 2 мы использовали конструкции owl, таким образом, включив эти ограничения в разрабатываемую онтологию. Некоторые из ограничений пункта 3 также можно указать в онтологии. Например, на owl можно записать то, что класс свойств-функций не пересекается (owl:disjoint) с классом свойств-констант (ограничение 3-a).

Также в онтологию можно включить модель соответствия указываемых состояний (ограничение 3-f). Это можно сделать, так как набор состояний вещества конечен и можно указать все валидные комбинации для указываемых состояний.

Остальные ограничения (3-b, 3-c, 3-d, 3-e) касаются не классов, а экземпляров свойств

(например, множество значений определяется для каждого свойства-функции отдельно), кроме того список свойств является открытым (список не фиксирован заранее, он может расширяться). Поэтому в разрабатываемой owl-онтологии, в которой содержатся лишь описания классов, такие ограничения смоделировать нельзя. Кроме того, в исходной базе не предусмотрена возможность задавать данные для этих ограничений (нет возможности хранить аргументы функций и т.д.)

Исходя из описанного выше, было решено в качестве эксперимента выбрать какое-либо из ограничений (3-b, 3-c, 3-d, 3-e) и придумать способ задавать его отдельно для экземпляров сущности «Свойство» (у нас используется класс `qudt:QuantityKind`).

Было решено взять ограничения 3-e (о множестве значений свойства-функции). Так как данное ограничение накладывается отдельно на каждый экземпляр класса `qudt:QuantityKind`, и при этом границы отрезка значений могут задаваться для разных единиц измерения, то необходимо требуемые ограничения формулировать для ресурсов внешнего словаря, который содержит все необходимые для этого данные. Поясним идею: наша исходная реляционная база призвана хранить информацию о значениях свойств веществ, однако детальная информация о каждой из используемых предметных областей не хранится (например, в нашей базе свойство может быть измерено только одной единицей измерения, а также нет параметров преобразования из одной единицы в другую и т.д.). Поэтому было бы разумнее взять существующий словарь свойств и единиц измерения на базе RDF, схема которого подробно моделирует их отношения и параметры, и задавать требуемые ограничения уже в терминах этого словаря. Как уже выяснено в ходе обзора, онтология QUDT подходит на роль такого внешнего источника-словаря. Поэтому будем формулировать ограничения на значения свойств, определенных в этой онтологии; ограничения будут выражены для единиц измерения, также определенных в QUDT.

Однако нам также потребуется как-то проверять сформулированные ограничения на наших данных, в которых экземпляры свойств определены отдельно от QUDT. Поэтому необходимо сделать привязку эквивалентности (предикат `owl:sameAs`) между нашими свойствами и свойствами, определенными в QUDT, рассматривая онтологию QUDT как внешний источник RDF ресурсов. Способы привязывания рассмотрим далее.

3.2 Преобразование исходных данных в RDF

Существует несколько инструментов для преобразования реляционных данных в формат RDF. Из бесплатных инструментов самым распространенным является платформа D2R [5]. Данный инструмент поддерживает работу с СУБД postgres.

Инструмент позволяет задать соответствие схемы реляционной базы на RDF-модель в виде т. н. маппинг-файла. Есть возможность ставить в соответствие одной таблице несколько классов сущностей, выборочно определять набор свойств для классов. В состав D2R платформы входит стандартное веб-приложение, которое позволяет просматривать полученные данные RDF в браузере, удобно навигируя по URL ресурсов. D2R предоставляет возможность доступа к данным по HTTP (просмотр данных-троек ресурса по его URL), а также с помощью SPARQL. Преобразование данных происходит на лету, то есть взаимодействие происходит непосредственно с данными из реляционной базы без сохранения в промежуточные хранилища. D2R позволяет провести все необходимые преобразования исходных данных, описанные выше. Данный инструмент подходит для преобразования и публикации исходных данных в RDF формате.

3.3 Подход к связыванию данных по свойствам веществ с внешними источниками.

D2R поддерживает лишь преобразование данных из реляционной базы в RDF, не предоставляя доступа к отдельным RDF-хранилищам, поэтому связи с внешними источниками тоже будем хранить в исходной реляционной базе, для чего нужно предусмотреть дополнительные таблицы.

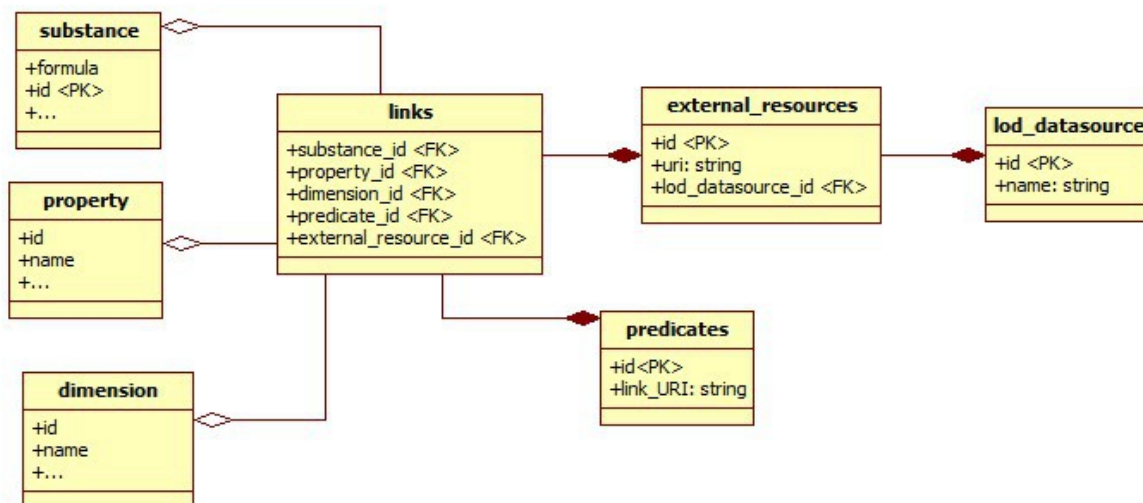


Рис. 18: Дополнительные таблицы для хранения связей

На рис. 18 изображены таблицы для хранения связей. Substance, property и dimension — таблицы, которые уже есть в исходной базе и хранят сущности, которые нужно будет связывать с внешними ресурсами. Остальные таблицы введены специально для хранения связей. Связи хранятся непосредственно в таблице links. В зависимости от того, сущность какого типа участвует в конкретной связи, указывается либо substance_id, либо property_id, либо dimension_id. Два оставшихся id поля должны быть равны null. Поле predicate_id указывает предикат для связи. Список предикатов хранится в таблице predicates. Поле external_resource_id указывает, к какому внешнему ресурсу проставлена привязка. Внешние ресурсы хранятся в таблице external_resources и могут принадлежать различным источникам данных в Linked Open Data (lod_daatasource). Таким образом в каждой записи таблицы links хранится тройка «вещество|единица измерения|свойство – предикат из списка — внешний ресурс». При преобразовании D2R эта запись должна превратиться в соответствующую RDF тройку, для чего нужно правильно составить маппинг D2R.

Связывание данных нужно отдельно продумать для сущностей-веществ и для сущностей-свойств и сущностей-единиц измерения. Рассмотрим отдельно две этих задачи.

3.3.1 Связывание с внешними источниками по веществам

Как было выяснено в ходе обзора химических данных в Linked Open Data, больше всего для связывания нам подходит набор ChemSpider. Кроме того данный набор использует

онтологию ChemAxiom, также выбранную нами для представления сущностей-веществ в нашем наборе.

Для сущности вещества в наших данных хранится систематическое название и формула. Эти критерии можно было бы использовать для автоматической генерации связей, однако в ходе экспериментов по автоматической генерации было выяснено, что систематические названия веществ в наборе ChemSpider не соответствуют названиям, принятым в нашем наборе. Кроме того молекулярная формула не идентифицирует однозначно вещество. Поэтому было решено реализовать пользовательский интерфейс привязывания сущностей веществ, определенных в нашем наборе к веществам ChemSpider. Интерфейс должен облегчать процесс нахождения соответствия между веществами нашего и внешнего наборов и генерировать соответствующие связи. Также должна быть предусмотрена функциональность удаления связей.

3.3.2 Связывание с внешними источниками по свойствам и единицам измерения и задание ограничений на численные значения свойств

Была обозначена необходимость генерировать связи между экземплярами единиц измерения и свойств, определенных у нас в наборе с экземплярами, определенными в QUDT. Здесь возникает несколько проблем :

- Отсутствуют критерии для автоматической генерации связей;
- Многих свойств и единиц измерения, которые используются у нас в наборе, не хватает в QUDT;

В связи с перечисленными проблемами рассмотрим следующий подход к процессу связывания по свойствам и единицам измерения. Возложим ответственность за связывание на пользователя, предоставив ему интерфейс связывания, а также предоставим интерфейс определения свойств и единиц измерения в терминах QUDT, и полученные экземпляры уже будем использовать как внешние ресурсы для связывания.

Также необходимо дать возможность пользователю определять ограничения на численные значения свойств для полученных экземпляров свойств и единиц измерения.

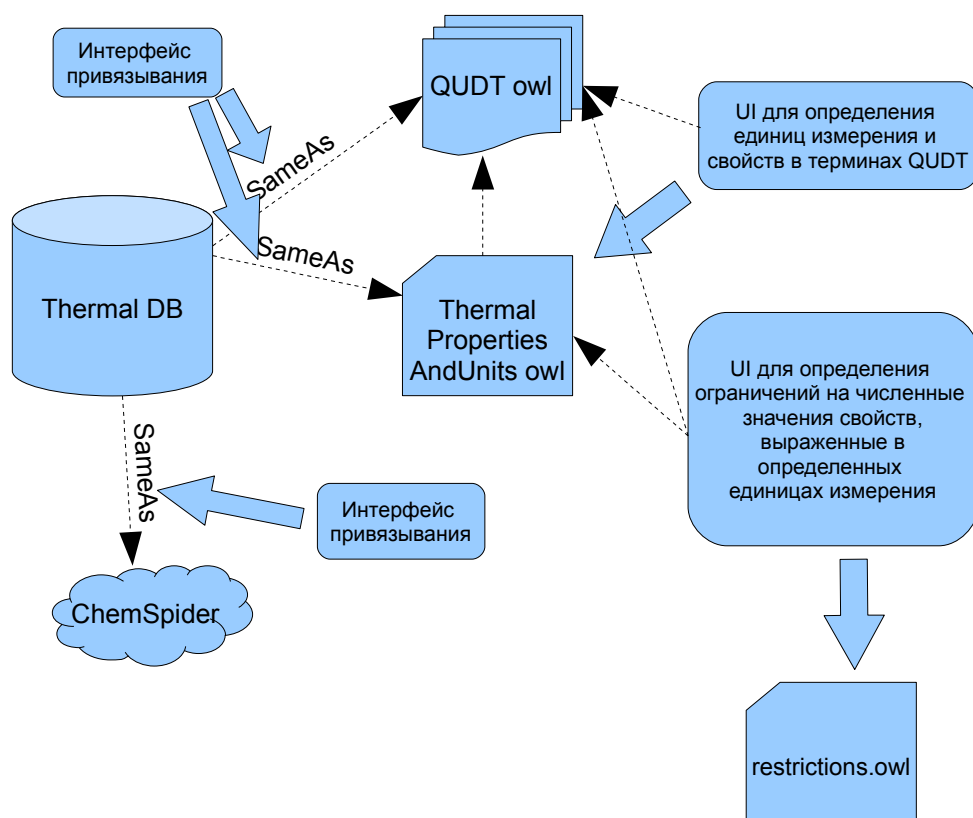


Рис. 19: Схема работы интерфейсов для осуществления связывания

На рис. 19 прямоугольниками с закругленными углами изображены пользовательские интерфейсы. Толстыми стрелками обозначены результаты использования интерфейсов, тонкими- исходные данные для работы с интерфейсами. Thermal DB — исходная база.

1. Интерфейс для определения свойств и единиц измерения в терминах QUDT должен предоставлять пользователю следующие возможности.
 - а) Просматривать экземпляры QUDT, и определять новые экземпляры на их основе.
В частности, данные интерфейс должен позволять :
 - i. указывать в качестве обобщения (qudt:generalization) нового свойства какое-то свойство уже определенное в QUDT;
 - ii. указывать для новой единицы измерения факт ее применимости к свойству, определенному в QUDT, либо созданному пользователем;
 - б) Редактировать и удалять определенные пользователем экземпляры свойств и единиц измерения;
 - с) В результате должна быть возможность сохранить в формате OWL созданные экземпляры;

2. Интерфейс привязывания для сущностей-веществ был обсужден выше, для свойств и единиц измерения требования аналогичные. Результатом работы интерфейса привязывания должна служить связь с внешним ресурсом, сохраненная в таблицах базы, введенных в пункте 3.3. На схеме связи обозначены как `sameAs`, так как будут использоваться предикаты `owl:sameAs`.
3. Интерфейс для определения ограничений на значения свойств веществ должен предоставлять пользователю следующие возможности:
 - а) Создание ограничений на значения свойств, определенных в терминах QUDT (либо в новой онтологии, созданной с помощью описанного выше интерфейса, либо непосредственно в QUDT);
 - б) Просмотр созданных ограничений;
 - в) Ограничения должны задаваться для определенной единицы измерения, применимой к свойству, для которого задается ограничение;
 - г) Ограничение на значение свойства имеет одну из форм
 - i. $[\min; \max]$,
 - ii. $(-\infty; \max]$,
 - iii. $[\min; +\infty)$, где \min и \max — минимальное и максимальное возможное значение данного свойства в данной единице измерения;
 - д) Если для свойства задается несколько ограничений, выраженных в разных единицах измерения, то полученные ограничения должны быть согласованы между собой. То есть при соответствующем переводе единиц измерения интервалы ограничений должны оказаться одинаковыми;
 - е) Должна быть возможность сохранить созданные ограничения в формате OWL;

Глава 4. Реализация интеграции данных по свойствам веществ в пространство Linked Open Data

В главе 3 был приведен детальный анализ задачи интеграции для конкретных данных по свойствам веществ в пространство Linked Open Data. Был раскрыт каждый из пунктов общей задачи интеграции.

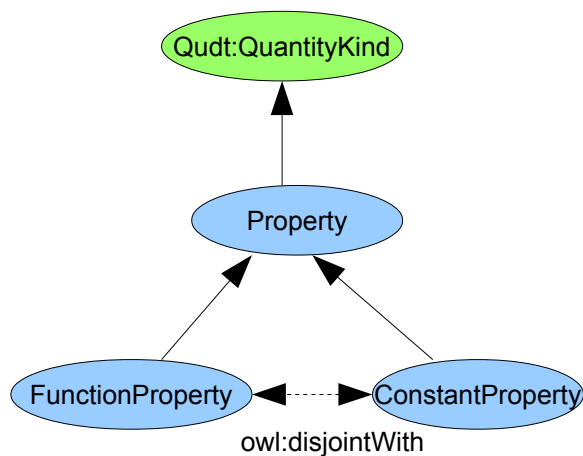
Теперь рассмотрим конкретную реализацию еще не рассмотренных моментов.

4.1 Модель ограничений предметной области в разрабатываемой онтологии.

В пункте 3.1.2 (Моделирование ограничений предметной области) прошлой главы были проанализированы ограничения, накладываемые на данные предметной областью. Все ограничения были разделены на 2 класса : а) ограничения, которые можно записать в моделируемой для предметной области онтологии, б) ограничения, которые нужно задавать отдельно для каждого экземпляра-свойства.

Реализация простейших ограничений была приведена. Рассмотрим принципы реализации в онтологии ограничений 3-а (непересекаемость классов свойств-функций и свойств-констант) и 3-f (согласованность указаний фазовых состояний) из пункта 3.1.2.

4.1.1 Непересекаемость классов свойств-функций и свойств-констант



В ходе построения онтологии в главе 3 было решено использовать класс `qudt:QuantityKind` для типизации экземпляров-свойств нашего набора.

Дополнительно введем класс `Property`, от которого унаследуем классы `FunctionProperty` (свойство-функция) и `ConstantFunction` (свойство-константа).

Сделаем эти два класса непересекающимися.

Приведем соответствующий фрагмент онтологии.

Property rdfs:subclass qudt:QuantityKind.

FunctionProperty rdfs:subclass qudt:Property.

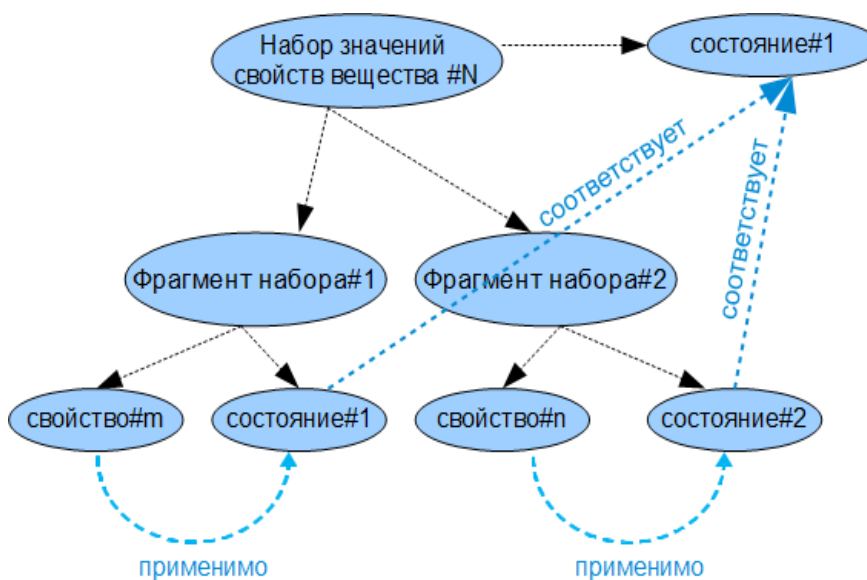
ConstantProperty rdfs:subclass qudt:Property.

FunctionProperty owl:disjointWith qudt:ConstantProperty.

Принцип валидации исходных данных.

Про исходные данные мы знаем, что значения свойств-функций хранятся в таблице data, а значения свойств-констант- в таблица constants_of_substance. В маппинг-файле D2R пропишем, что будем классифицировать свойства, на которые есть ссылки из таблицы data, как свойства-функции; а те свойства, на которые есть ссылки из таблицы constants_of_substance — как свойства-константы. Если в базе окажутся две записи в таблицах data и constants_of_substance, которые ссылаются на одно и то же свойство, то D2R припишет этому свойству принадлежность сразу к двум классам FunctionProperty и ConstantProperty, которые по определению в онтологии не должны иметь общих экземпляров. Онтология и данные окажутся противоречивы по логике OWL.

4.1.2 Согласованность ссылок на состояния вещества при определении наборов данных, фрагментов данных и применимости свойств-функций к состояниям



Ссылка на состояние вещества есть в таблицах environment_conditions, data и property. После преобразования в RDF эта ссылка должна быть атрибутом экземпляров классов Dataset, Data и Property.

На рис. 21 показан пример фрагмента исходных данных в терминах разработанной

онтологии.

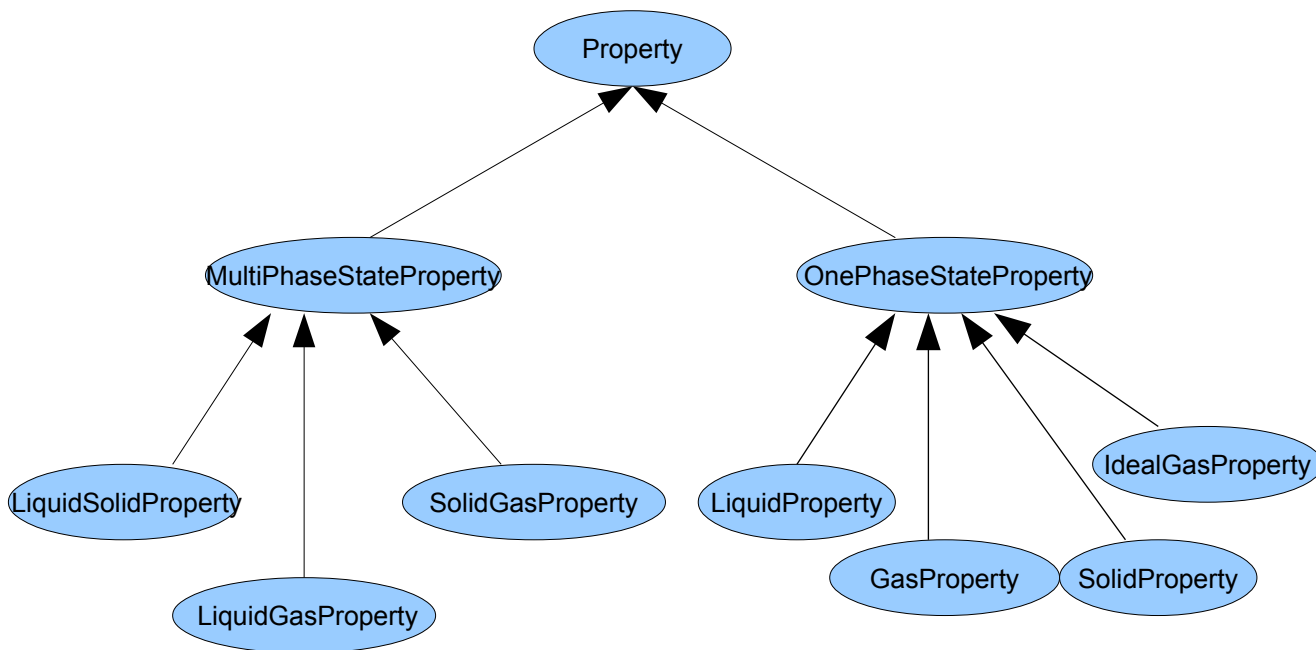
Набор данных#N вещества (экземпляр класса Dataset в нашей онтологии) состоит из двух фрагментов данных #1 и #2 (экземпляры класса Data), которые задаются для свойств #m и #n.

Набор данных рассматривается для определенного фазового состояния вещества. Свойство может быть применимо к одному или нескольким состояниям, а фрагменты данных содержат ссылку на состояние на тот случай, если состояние, указанное для набора является составным.

Итак, требуется чтобы указанные состояния были согласованны. Формализуем задачу моделирования:

- Состояние, указанное в фрагменте набора численных значений свойств для вещества должно соответствовать состоянию, указанному для набора :
 - то есть совпадать с состоянием для набора или «входить» в него. (Например, состояние «газ» входит в состояние «газ-жидкость»).
- Свойство, указанное в фрагменте набора численных значений свойств для веществ должно быть применимо к состоянию, указанному в данном фрагменте.

Для того, чтобы сформулировать данную модель на языке OWL, введем подклассы Dataset, Data и Property, соответствующие указанным состояниям. То есть вместо того, чтобы присваивать экземплярам свойство со значением состояния, мы будем присваивать им соответствующие классы. На рис. 22 изображена классификация свойств веществ по применимости к различным фазовым состояниям вещества.



Свойство может быть применимо к нескольким состояниям, следовательно, оно может относиться сразу к нескольким из перечисленных классов.

Аналогичную классификацию введем для Dataset и Data.

Требуемые условия согласованности теперь можно будет выразить через введенные классы. Например, можно записать на owl, что LiquidData может ссылаться только на LiquidProperty и т.д. При этом нужно также объявить, что все подклассы Data нижних уровней не пересекаются(owl:disjointWith).

```
LiquidData rdfs:subClassOf Restriction [  
  owl:onProperty qudt:quantityKind.  
  owl:allValuesFrom LiquidProperty  
]
```

Аналогично сформулируем все остальные ограничения на комбинации Data-Property. Схема иерархии показана в приложении С.

Для Dataset тоже введем аналогичную иерархию. Введем также ограничения на фрагменты данных, являющиеся частью наборов данных (Dataset). Например, для набора данных вещества в газовом состоянии (GasDataset) — все фрагменты данных должны быть также указаны в газовом состоянии(GasData).

```
GasDataset rdfs:subClassOf Restriction [  
  owl:onProperty data.  
  owl:allValuesFrom GasDataset.  
]
```

Для набора данных вещества в состоянии равновесия «газ-жидкость» (LiquidGasDataset) все фрагменты данных могут быть указаны либо в состоянии газа, либо в состоянии жидкости, либо в состоянии «газ-жидкость».

```
LiquidGasDataset rdfs:subClassOf Restriction [  
  owl:onProperty data.  
  owl:allValuesFrom owl:Union [GasData, LiquidData, LiquidGasData].  
]
```

Аналогично запишем ограничения на все остальные комбинации подклассов Dataset и Data. Схема иерархии подклассов Dataset указана в приложении D.

Принцип валидации исходных данных

Зададим в маппинг-файле D2R классификацию для экземпляров Dataset, Data, Property в зависимости от значения ссылки на состояние вещества (это делается с помощью конструкции Translation Table, ставящей в соответствие литеральным значениям предопределенные URI, записываемые в маппинг-файле). Таким образом, в случае, если в исходной базе ссылки на состояния были проставлены неправильно, то не будут соблюдены ограничения, описанные выше.

Например, если в numerical_conditions стояла ссылка на состояние «газ», а в одной из соответствующих записях в data — на состояние «жидкость», то при преобразовании в RDF получится что у набора данных вещества в газовом состоянии (GasDataset) есть фрагмент в жидком состоянии (LiquidData). Поскольку класс LiquidData не пересекается с классом GasData, то такие данные будут неконсистентными по отношению к описанной модели в логике OWL.

Здесь важным моментом является то, что мы сказали, что GasData и LiquidData — непересекающиеся классы (иначе соответствующий экземпляр Data был бы отнесен по логике owl также и к классу GasData и противоречия онтологии не было бы).

Однако для аналогичных подклассов Property мы не можем задать непересекаемости классов, так как свойство может быть применимо к нескольким состояниям. Поэтому введем для каждого из подклассов Property дополнительный несовместимый класс. Для LiquidProperty-NotLiquidProperty, для GasProperty – NotGasProperty и т. д. Элементы каждой такой пары объявим непересекающимися классами. В маппинг-файле D2R пропишем, что свойства, для которых не описана применимость к одному из состояний — неприменимы к этому состоянию (то есть принадлежат к дополнительному классу, соответствующему рассматриваемому состоянию). Это позволит нам проводить валидацию комбинации «Data-Property». Если для записи в таблице data было указано состояние «газ», а свойство (property_id) для этой записи в data — не применимо к этому состоянию, то при преобразовании в RDF получится, что экземпляр GasData ссылается с помощью предиката qudt:quantityKind на экземпляр NotGasProperty, что является нарушением ограничений описанных выше.

4.2 Создание новых экземпляров-свойств и единиц измерения в терминах QUDT и задание ограничений на численные значения свойств

В соответствии с требованиями, сформулированными в главе 3, были реализованы интерфейсы

1. определения новых экземпляров свойств и единиц измерения в терминах QUDT,
2. задания ограничений на численные значения этих свойств.

Требуемые интерфейсы были реализованы в виде веб-приложения, написанного на языке Java с использованием библиотеки jena [12] для работы с RDF-данными и моделью OWL.

Был выбран вариант написать web-приложение, так как необходимо обеспечить возможность удаленной работы пользователя с приложением. Jena является удобной и мощной библиотекой для работы с RDF-данными и OWL-онтологиями, предоставляет интерфейсы для создания элементов онтологий (классы, предикаты, разные типы Restriction-ограничений, и т.д.), а также удобные возможности по вводу-выводу owl онтологий. Во время работы приложения все онтологии содержатся в оперативной памяти.

4.2.1 Реализованные интерфейсы

Ontologies

New Quantity&Unit Complement Ontology

Select file to upload statements to new ontology (complement to QUDT). All statements of current qudt complement ontology will be erased: Файл не выбран

[Download new ontology](#)

[clear current new ontology](#)

Restriction ontology

Select file to upload statements to restriction ontology . All statements of current restriction ontology will be erased: Файл не выбран

[Download restriction ontology](#)

[clear current restriction ontology](#)

Рис. 22: Загрузка/выгрузка онтологий

На рис. 22 изображен фрагмент интерфейса, отвечающий за загрузку/выгрузку онтологий свойств и единиц измерения (New Quantity&Unit Complement Ontology) и ограничений на значения свойств (Restriction Ontology). Пользователь может загрузить готовую ранее сохраненную онтологию и продолжить работать с ней или начать с пустой онтологией. После окончания работы пользователь может выгрузить полученную онтологию.

Navigation

[Unit class hierarchy](#)

[Quantity kind class hierarchy](#)

Current QUDT class

[Create new instance of this type](#)

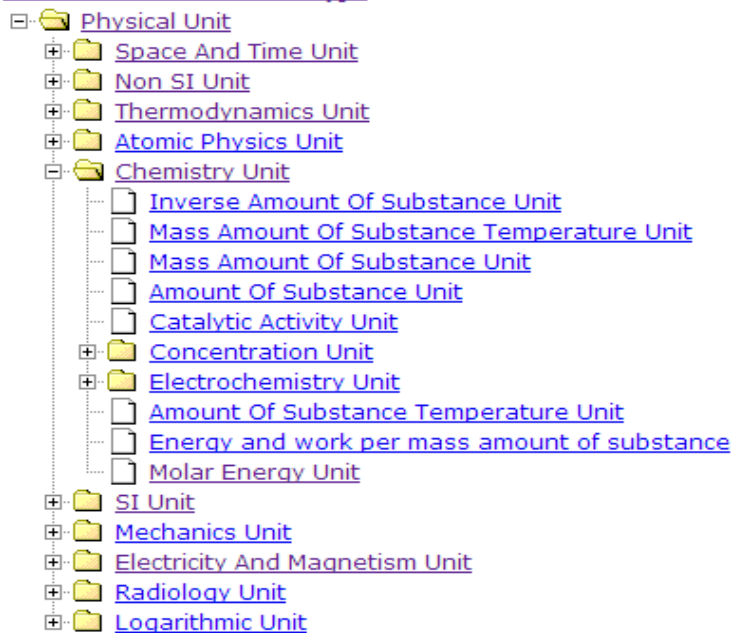


Рис. 23: Навигация по категориям единиц измерения

На рис. 23 показана навигация по категориям единиц измерения в приложении. Если перейти по ссылке «Quantity kind class hierarchy», то можно увидеть аналогичную иерархию по категориям измеряемых свойств.

Current QUDT class

[Create new instance of this type](#)

☐ [Chemistry](#)

This class direct instances

Qudt ontology instances

[Concentration](#)
[Energy and Work per Mass Amount of Substance](#)
[Length Molar Energy](#)
[Turbidity](#)
[Inverse Amount of Substance](#)
[Electric Charge per Amount of Substance](#)
[Amount of Substance per Unit Mass](#)
[Amount of Substance Per Unit Volume](#)
[Molecular Mass](#)
[Amount of Substance](#)
[Mass Amount of Substance](#)
[Mole Fraction](#)
[Catalytic Activity](#)
[Temperature Amount of Substance](#)
[Molar Volume](#)
[Molar Energy](#)
[Mass Amount of Substance Temperature](#)
[Molar Mass](#)

New ontology instances

[enthalpy of vaporization](#)

Рис. 24: Список химических свойств

ontology instances), и список свойств, определенных в новой пользовательской онтологии (New ontology instances).

На рис. 25 приведена страница экземпляра свойства. Только экземпляры, определенные в пользовательской онтологии можно редактировать и удалять (ссылки «remove», «edit»). На этой странице можно просмотреть список ограничений, заданных для данного свойства («restrictions on this quantity kind»), список единиц измерения, применимых к данному свойству («units applicable to this quantity kind»). Обобщением данного свойства (generalization) является свойство MolarEnergy, определенное в QUDT, поэтому список применимых единиц измерения содержит все единицы измерения, применимые также и к Molar Energy.

Current Instance

[edit](#)

[remove](#)

type [ChemistryQuantityKind](#)
uri <http://thermal#EnthalpyOfVaporization>
label enthalpy of vaporization
generalization [MolarEnergy](#)
description enthalpy of vaporization
[Create restriction for this quantity kind](#)

Restrictions on this quantity kind :

[Restriction for value in unit http://thermal#KiloJoulePerMole](#)

Units applicable to this quantity kind:

[KiloJoulePerMole](#)
[KilocaloriePerMole](#)
[JoulePerMole](#)

Рис. 25: Страница просмотра экземпляра свойства

Рис. 24 демонстрирует списки свойств, определенных в онтологии QUDT и в нашей новой онтологии в категории химических свойств (Chemistry). Приводятся отдельно список свойств, определенных в QUDT (Qudt

Можно перейти на страницу создания нового ограничения на значения данного свойства («create restriction for this quantity kind»).



Delete this restriction

quantityKind : <http://thermal#EnthalpyOfVaporization>

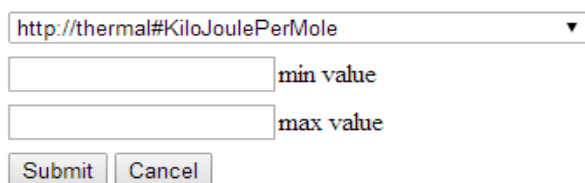
unit : <http://thermal#KiloJoulePerMole>

min : 0.0

max : 10000.0

Рис. 26 демонстрирует страницу просмотра ограничения на значения свойства «EnthalpyOfVaporization», выраженного в единицах «KiloJoulePerMole». Ограничение можно удалить.

Рис. 26: страница ограничения на значения свойства



<http://thermal#KiloJoulePerMole>

min value

max value

Submit Cancel

При создании ограничения пользователю предлагаются применимые единицы измерения, относительно которых можно задать максимум и минимум значений свойства (рис. 27).

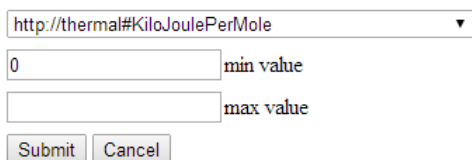
Рис. 27: создание ограничения на значения свойства

Если на данное свойство еще не создано ни одного ограничения, то пользователю

предлагается возможность ввести максимум и минимум. Если одно из чисел min value или max value не было введено, то считается что соответствующей нижней или верхней границы ограничения не задано. Должно быть задано одно из чисел min value или max value.

Если у единицы измерения не указаны параметры перевода единиц, то ограничение, выраженное через эту единицу, создать нельзя (рис. 28).

cannot create restriction on this unit : unit <http://data.nasa.gov/qudt/owl/unit#kilocaloriepermole> does not have conversion offset or multiplier



<http://thermal#KiloJoulePerMole>

0 min value

max value

Submit Cancel

Рис. 28: Система выдает сообщение о невозможности создать ограничение для единицы измерения, для которой не указаны параметры перевода

Рис. 29 : Страница создания ограничения на значения свойства, для которого уже заданы ограничения

Если в системе уже создано ограничение на значения данного свойства, то пользователю предлагается выбрать только единицу измерения (рис. 29), числа min value и max value система рассчитает автоматически, произведя перевод границ интервала уже существующего

ограничения.

4.2.2 Принцип работы создаваемых ограничений на значения свойств

Принцип работы ограничений заключается в том, что для конкретной пары «Свойство-Единица измерения» создается «класс-ловец», в который по логике owl попадают все экземпляры-численные значения (qudt:QuantityValue), заданные для данного свойства через данную единицу измерения. Для данного класса определяется класс-родитель (rdfs:subClassOf) «валидатор», на который накладываются ограничения на численное значение свойства qudt:numericValue данного экземпляра класса qudt:QuantityValue(конструкции owl minInclusive, maxInclusive). По правилам логического вывода OWL соответствующие экземпляры QuantityValue будут отнесены к созданному классу-ловцу, и как следствие — к классу валидатору. В случае нарушения ограничения на численное значение (условия принадлежности к классу-валидатору), проверяемое численное значение будет неконсистентным по логике OWL по отношению к системе из класса-ловца и класса-валидатора.

(Данный экземпляр, будучи отнесенным к классу «ловец», по определению rdfs:subClassOf должен принадлежать классу родителю «валидатору», но из-за несоблюдения условий на численное значение — не принадлежит. Это противоречие.)

Схема примера такого ограничения, записанного на owl, представлена в приложении Е.

4.3 Реализация связывания сущностей-веществ из исходных данных с ресурсами ChemSpider

В данный момент идут независимые работы по созданию веб-приложения для работы с

исходными данными в реляционной базе. Было решено встроить требуемый интерфейс привязывания по веществам в данное приложение.

Реализованный интерфейс использует http-точку поиска ChemSpider для того, чтобы искать вещества по формуле и предлагать пользователю варианты для связывания.

ChemSpider substances with the same formula listing :


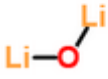

ChemSpider Id	ChemSpider Image	ChemSpider Systematic Name	ChemSpider Formula
145811		dilithium;oxygen(2-)	Li2O
19484298			Li2O
26563443		lithium(1+) dihydride; oxygen(2-)	Li2O

Рис. 30: Список веществ ChemSpider, предоставляемый пользователю

owl:sameAs – URI_вещества_из_ChemSpider». Вся информация о веществах, отображаемая на странице, забирается из ChemSpider по протоколу HTTP.

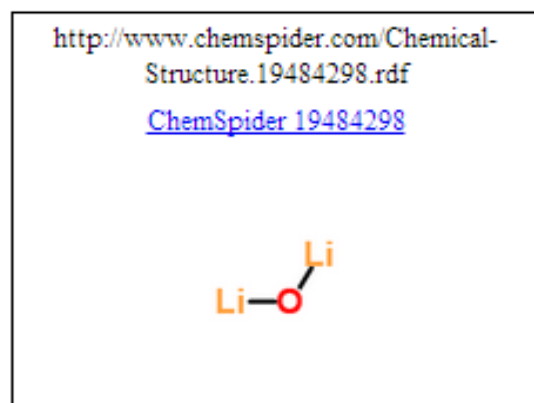
На рис. 31 показана отображенная в интерфейсе созданная привязка к веществу ChemSpider. Пользователь может удалить созданную привязку или привязать другое вещество вместо привязанного в данный момент.

На рис. 30 показана страница выбора вещества ChemSpider. На нее пользователь попадает со страницы определенного вещества, для которого нужно найти соответствующее в ChemSpider вещество. Пользователь может выбрать одно из предложенных веществ и нажать кнопку Bind. В результате будет создана и сохранена в базу соответствующая связь, которая при преобразовании в RDF, задаваемым написанным маппингом-файлом D2R, превратится в тройку вида

«URI_нашего_вещества —

[Найти соответствующее вещество в ChemSpider](#)

This substance has a link to ChemSpider



[Удалить привязку](#)

Рис. 31 : Созданная связь с ChemSpider на странице вещества

Заключение

В ходе выполнения дипломной работы была проанализирована задача интеграции в пространство Linked Open Data данных по теплофизическим свойствам веществ. Был сделан обзор тематически близких источников данных в Linked Open Data, а также онтологий смежных предметных областей. На основе результатов работы [2] и проведенного обзора была составлена онтология на языке OWL, использующая термины онтологий ChemAxiom и QUDT. Исходные данные были представлены в формате RDF с помощью инструмента D2R.

Был выявлен источник ChemSpider, пригодный для связи по экземплярам-веществам. Для генерации соответствующих связей был реализован пользовательский веб-интерфейс, облегчающий процесс выбора подходящего для связи вещества ChemSpider и записи сохраненных связей в модифицированную исходную базу.

Был проведен анализ ограничений предметной области. Те ограничения, которые касаются классов данных, были сформулированы в разработанной OWL-онтологии предметной области. Также были рассмотрены ограничения, которые касаются использования экземпляров классов. В том числе был разработан способ задания с помощью OWL ограничений на численные значения свойств веществ, выраженные в определенных единицах измерения. Для этого было разработано веб-приложение, позволяющее задавать новые свойства и единицы измерения в терминах QUDT и задавать ограничения на численные значения этих свойств.

Реализация интерфейса привязывания экземпляров свойств и единиц измерения из нашей базы к экземплярам свойств и единиц измерения, заданным отдельно (в онтологии QUDT или онтологии, созданной с помощью упомянутого выше интерфейса), остается в качестве дальнейшей работы над данной темой. Разработка способов задания других ограничений предметной области также остается в качестве направления дальнейших исследований.

Используемые термины

Semantic Web (Семантическая Паутина) - направление развития Всемирной паутины (WWW), целью которого является предоставление информации в виде, пригодном для машинной обработки, то есть представление в структурированном виде. Концепция Semantic Web была принята Консорциумом Всемирной паутины (W3C), а для ее поддержки были введены стандарты представления данных (RDF – Resource Description Framework) и описания схем (*онтологий*) на эти данные (RDFS, OWL).

Основной идеей Semantic Web является реализация архитектуры Web , которая могла бы обеспечить переиспользование структурированных данных в глобальном масштабе.

URI (англ. Uniform Resource Identifier, рус. Единообразный идентификатор ресурса)— это последовательность символов, идентифицирующая абстрактный или физический ресурс. ***Одним из вариантов URI является URL (англ. Uniform Resource Locator, рус. Единообразный локатор ресурса)*** — это стандартизированный способ записи адреса ресурса в сети Интернет.

Ресурс — единица описываемых данных в Semantic Web. Каждый ресурс обозначает какой-либо реальный объект, понятие или явление и имеет идентификатор URI (Unified Resource Identifier), который используется для описания знаний о данной сущности.

RDF (Resource Description Framework) – разработанная консорциумом Всемирной Паутины модель для представления данных в виде троек "субъект - предикат - объект", где субъект - это описываемый ресурс, предикат - типизированная связь (термин для обозначения этой типизированной связи должен быть определен в используемой онтологии), объект - литерал или другой ресурс, объект считается «значением» свойства, определенного предикатом для ресурса-субъекта. RDF определяет вспомогательный словарь терминов для представления данных.

Онтология в контексте Semantic Web - это схема определенного набора данных, которая предоставляет описание классов данных и связей между ними, а также определяет некоторые ограничения на использование указанных классов и связей.

RDFS (RDF Schema) — словарь терминов для моделирования RDF данных, расширяющий возможности RDF.

SPARQL (рекурсивный акроним от англ. SPARQL Protocol and RDF Query Language) — язык запросов к данным, представленным по модели RDF, а также протокол для передачи этих запросов и ответов на них.

OWL (Web Ontology Language) — стандарт для записей онтологий в контексте Semantic Web, расширение RDF, определяет некоторые связи и отношения, не предусмотренные в RDFS.

Linked Data — термин, описывающий набор практик, рекомендованных для публикации и связывания структурированных данных. Также этот термин может использоваться для описания данных, опубликованных в соответствии с указанным набором практик.

Linked Open Data (LOD) — термин, описывающий ту часть данных Linked Data, которая находится в свободном доступе.

Основные принципы Linked Data — принципы, сформулированные для публикации данных в Linked Open Data [7]:

- Использование URI (Unified Resource Identifier) для идентификации ресурсов;
- Использование HTTP URIs для предоставления доступа к описанию ресурсов;
- Представление информации в соответствии со стандартами Semantic Web (RDF, SPARQL) ;
- Включать в описание ресурсов ссылки на другие ресурсы, определенные в Linked Data.

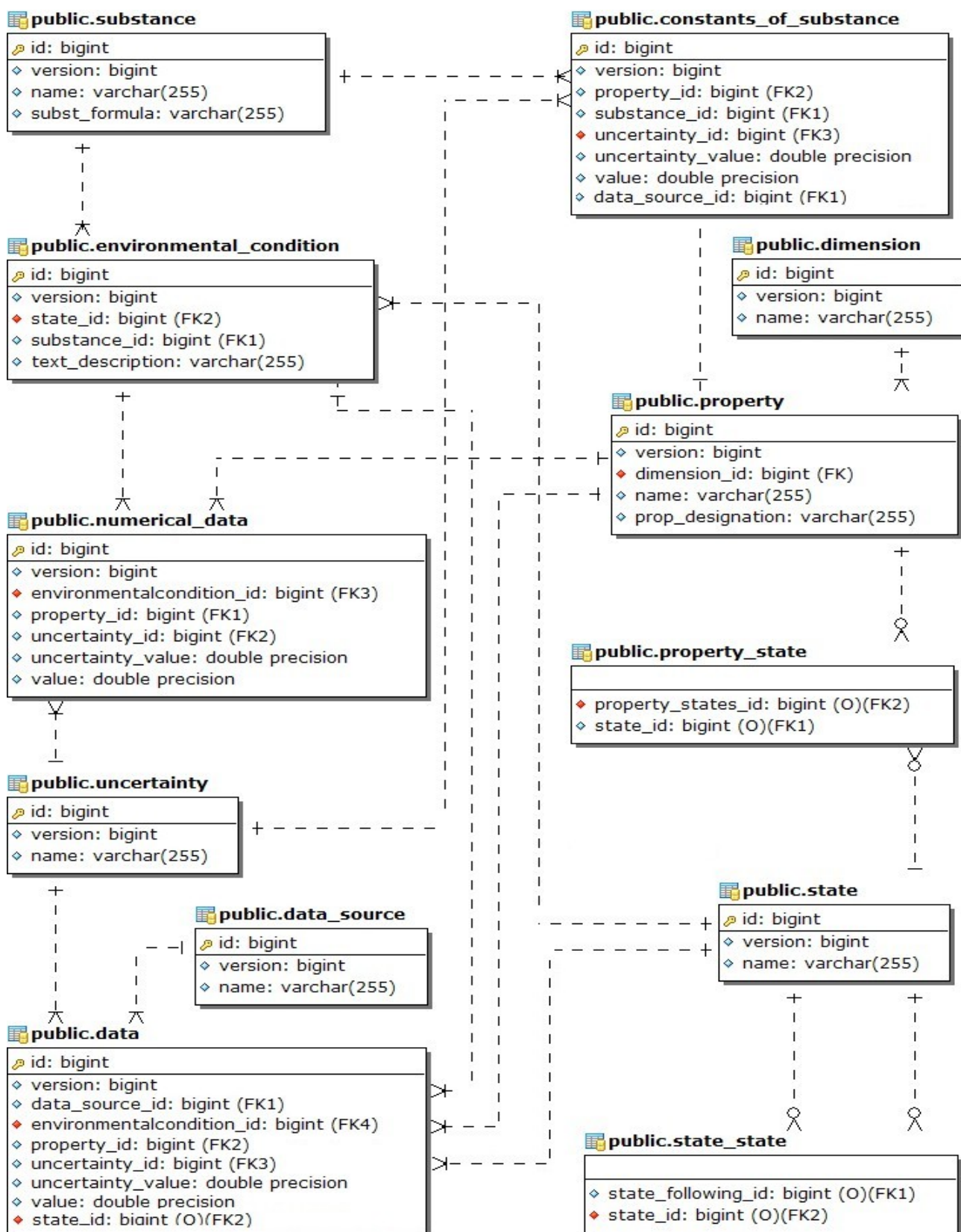
Список литературы

- [1] А.О. Еркимбаев, В.Ю. Зицерман, Г.А. Кобзев, В.А. Серебряков, Л.Н. Шиолашвили Интеграция данных по свойствам веществ и материалов на основе онтологического моделирования предметной области // Труды XV Всероссийской объединенной конференции «Интернет и современное общество» (IMS-2012). Санкт-Петербург, 2012. С. 38-47.
- [2] О.М. Атаева, А.О. Еркимбаев, В.Ю. Зицерман, Г.А. Кобзев, В.А. Серебряков, К.Б. Теймуразов, Хайруллин Р.И. Интеграция данных по теплофизическим свойствам веществ методами онтологического моделирования // Труды 15-й Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» (RCDL-2013). Ярославль, 2013. С. 43-51.
- [3] Hodgson R, Keller PJ QUDT - Quantities, Units, Dimensions and Data Types [HTML] (<http://qudt.org/>).
- [4] N. Adams, E.O. Cannon, P. Murray-Rust ChemAxiom – An ontological framework for chemistry in Science // Nature precedings, 2009.
(<http://precedings.nature.com/documents/3714/version/1>).
- [5] D2RQ platform -Accessing relational databases as virtual RDF graphs [HTML] (<http://d2rq.org/>).
- [6] ChemSpider - Search and share chemistry [HTML] (<http://rdf.chemspider.com/>).
- [7] Tom Heath , Christian Bizer Linked Data: Evolving the Web into a Global Data Space [HTML] (<http://linkeddatatoolkit.com/editions/1.0/>).
- [8] Dean Allemang, James Hendler Semantic Web for working ontologist. 225 Wyman Street, Waltham, USA : Morgan Kaufmann Publishers, 2011. 354 p.
- [9] Brian McBride, Dan Brickley, R.V. Guha RDF Schema 1.1 W3C Recommendation 25 February 2014 [HTML] (<http://www.w3.org/TR/rdf-schema/>).
- [10] Sean Bechhofer, Frank van Harmelen, Jim Hendler, Ian Horrocks, Deborah L. McGuinness, Peter F. Patel-Schneider, Lynn Andrea Stein OWL Web Ontology Language Reference [HTML] (<http://www.w3.org/TR/owl-ref/>).
- [11] Egon L Willighagen¹, Andra Waagmeester¹, Ola Spjuth, Peter Ansell, Antony J Williams, Valery Tkachenko, Janna Hastings, Bin Chen and David J Wild The ChEMBL

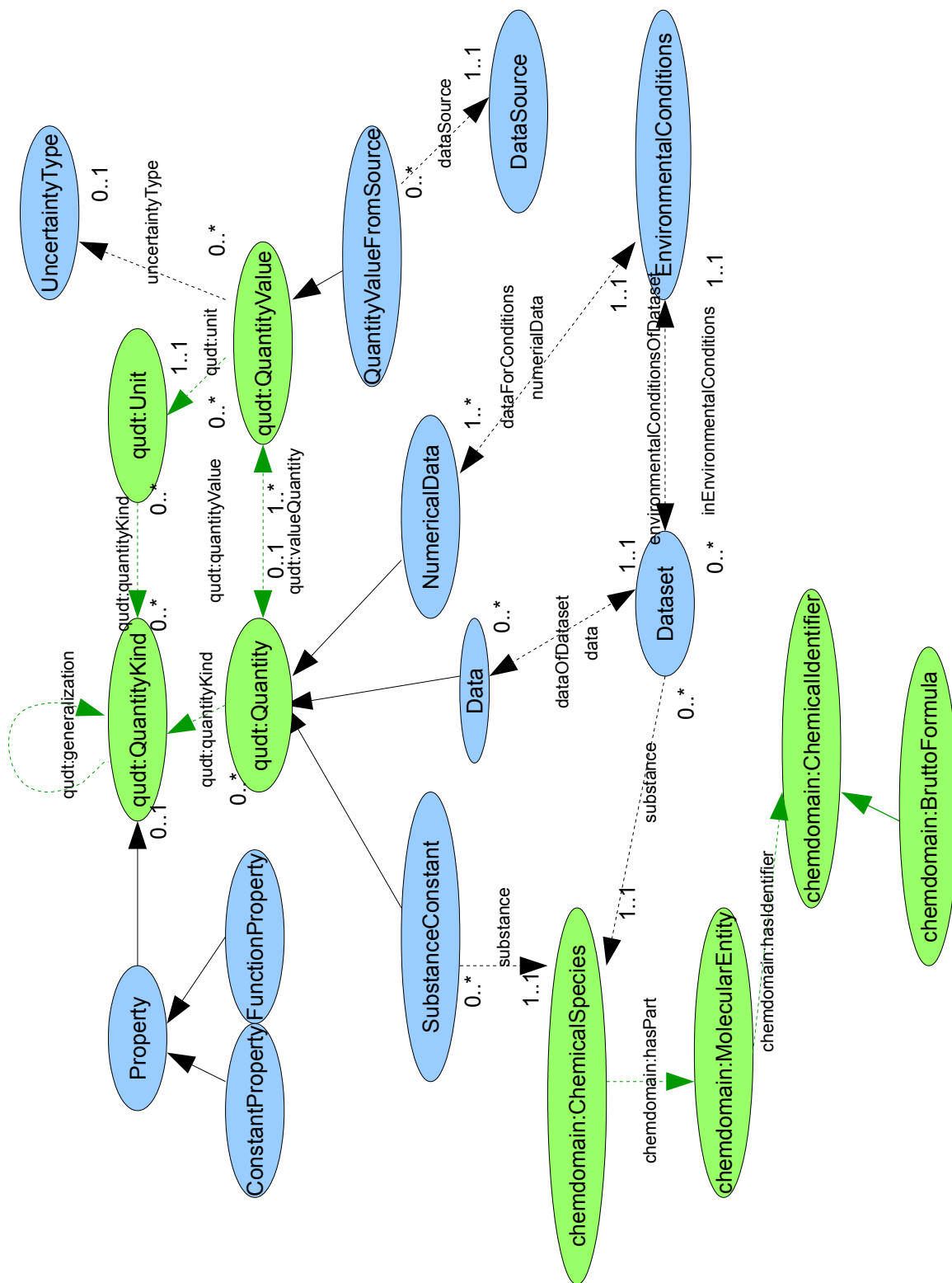
database as linked open data // Journal of Cheminformatics. 2013. [PDF]
(<http://www.jcheminf.com/content/5/1/23>).

- [12] Apache Jena A free and open source Java framework for building Semantic Web and Linked Data applications [HTML] (<http://jena.apache.org/>).

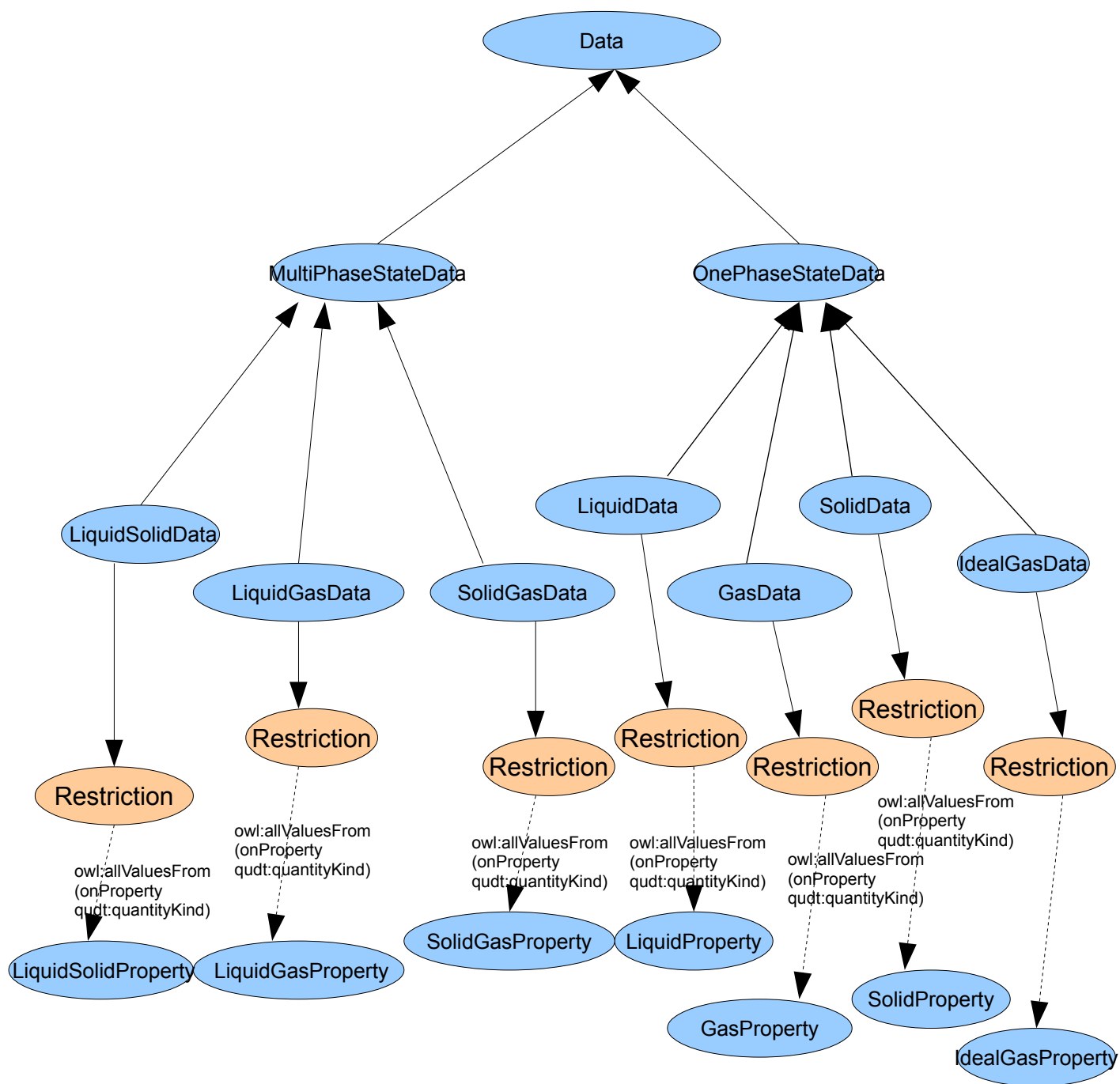
Приложение А. Схема исходной реляционной базы данных



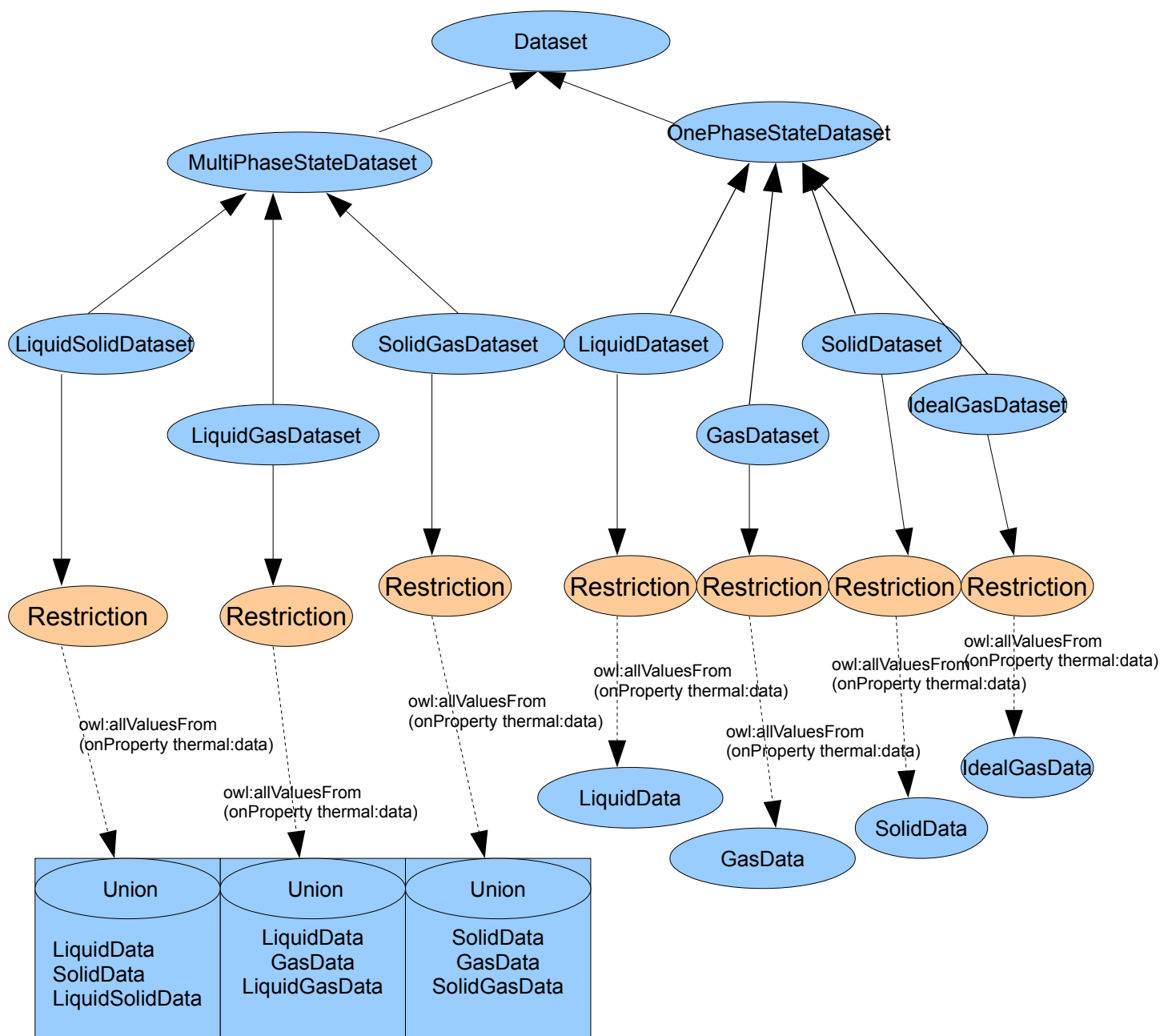
Приложение В. Общая схема разработанной онтологии



Приложение С. Иерархия подклассов Data



Приложение D. Иерархия подклассов Dataset



Приложение Е. Схема представления ограничений на численные значения свойств (определенных в терминах онтологии QUDT), записанных на языке OWL

