САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Математико-механический факультет

Кафедра системного программирования

РАЗРАБОТКА И РЕАЛИЗАЦИЯ МЕТОДОВ И ИНСТРУМЕНТАРИЯ ДЛЯ СРАВНЕНИЯ И ОБЪЕДИНЕНИЯ ОНТОЛОГИЙ

Дипломная работа студентки 545 группы

Жуковой Анны Руслановны

Научный руководитель	…… / подпись /	к. ф.-м. н. Бугайченко Д. Ю.
Рецензент	…… / подпись /	доцент Соловьев И. П.
“Допустить к защите” заведующий кафедрой,	…… / подпись /	д.ф.-м.н., проф. Терехов А. Н.

Санкт-Петербург
2010

SAINT PETERSBURG STATE UNIVERSITY
Mathematics & Mechanics Faculty

Software Engineering Chair

DEVELOPMENT OF ONTOLOGY COMPARISON AND MAPPING METHOD AND TOOL

by

Anna Zhukova

Master's thesis

Supervisor	……	PhD D. Y. Bugaychenko
Reviewer	……	Ass. Prof. I. P. Soloviev
"Approved by" Head of Department	……	Professor A. N. Terekhov

Saint Petersburg
2010

Оглавление

Введение

Основные понятия

С распространением Всемирной паутины, идея информационной сети, доступной по всему миру, стала повсеместной. Всемирная паутина основывается на идее открытого сообщества: каждый может внести свой вклад, поделившись своими идеями и информацией, и эта информация станет доступной всему сообществу. Результатом такой открытости стала потрясающая освещенность практически любого вопроса. Появилось огромное число веб-сервисов и автономных агентов, их использующих.

Однако, оборотной стороной такой полноты информации стала сложность нахождения в данном огромном разнообразии ответов на конкретный вопрос. Одной из причин такой проблемы является то, что основным форматом данных ресурсов Всемирной паутины в настоящий момент является текст. Текстовый анализ документов не слишком эффективен, текст многозначен и неудобен для автоматической обработки. Одни и те же слова могут относиться к совершенно разным понятиям: например, словосочетание «Шерлок Холмс» может встречаться как в документе, содержащем информацию о книжном персонаже, так и в описании одноименного паба в Лондоне.

В ходе работы над проблемой такой неоднозначности был разработана концепция Семантической паутины.

Семантическая паутина (Semantic Web) [18.] — часть глобальной концепции развития сети Интернет, целью которой является реализация возможности машинной обработки информации, доступной во Всемирной паутине. Основной акцент концепции делается на работе с метаданными, однозначно характеризующими свойства и содержание ресурсов Всемирной паутины,

вместо используемого в настоящее время текстового анализа документов. В семантической паутине предполагается повсеместное использование, во-первых, унифицированных идентификаторов ресурсов (URI), а во-вторых — онтологий и языков описания метаданных.

Для аннотирования ресурсов в Семантической паутине применяются онтологии.

Онтология [3.] — это попытка всеобъемлющей и детальной формализации некоторой области знаний с помощью концептуальной схемы. Обычно такая схема состоит из структуры данных, содержащей все релевантные классы объектов, их связи и правила (теоремы, ограничения), принятые в этой области. Онтологии используются в процессе программирования как форма представления знаний о реальном мире или его части.

За счет введение метаданных и аннотирования с их помощью ресурсов Всемирной паутины, можно добиться однозначности: вместо многозначного текста в такой модели рассматриваются уникальные идентификаторы URI.

Но такой подход вносит новую проблему. Вследствие неуникального именования[1.], одному и тому же понятию могут соответсвовать несколько URI, например, взятых из разных онтологий. Таким образом, уйдя от проблемы многозначности текстового подхода и задачи фильтрации результатов, выдаваемых по текстовому запросу с тем, чтобы оставить лишь соотвествующие искомому значению, в модели Семантической паутины мы переходим к проблеме агрегации: сопоставления ресурсов, помеченных разными URI, но имеющих один и тот же смысл.

При нахождения лишь в рамках, ограниченных одной онтологией: когда все данные проаннотированы понятиями одной и той же онтологии, эта проблема отсуствует. Но на практике такое ограничение недостижимо даже в очень узких областях. Даже если система использует всего одну онтологию, в процессе своего развития, и, соотвественно, модификаций онтологии, она

может столкнуться с задачей сопоставления аннотаций, соотвествующих разным версиям этой онтологии.

В более общем случае возникает задача сопоставления разных онтологий, нахождения общих для них понятий и часто объединения найденных онтологий в одну.

Задача сравнения актуальна и при создании новой онтологии: при попытке поиска уже существующих сходных онтологий для того, чтобы переиспользовать имеющуюся в них информацию, вместо создания ее с нуля (и заодно вместо увеличения числа неуникальных именований).

Наиболее остро проблема сравнения и сопоставления онтологий стоит в работе с интеллектуальными агентами.

В искусственном интеллекте, под термином интеллектуальный агент [9.] понимаются разумные сущности, наблюдающие за окружающей средой и действующие в ней, при этом их поведение рационально в том смысле, что они способны к пониманию и их действия всегда направлены на достижение какой-либо цели. Такой агент может быть как роботом, так и встроенной программной системой. Об интеллектуальности агента можно говорить, если он взаимодействует с окружающей средой примерно так же, как действовал бы человек. Одним из примеров заданий, выполняемых агентами, может служить задача постоянного поиска и сбора необходимой информации в Интернете.

Во время своей работы, интеллектуальному агенту приходится сравнивать имеющиеся у него онтологии с новыми, найденными в сети, а также с онтологиями других интеллектуалных агентов, с которыми он пытается взаимодействовать.

Постановка задачи

1. Провести сравнительный анализ существующих методов и инструментов сравнения и объединения онтологий.
2. Разработать метод сравнения онтологий, позволяющий численно оценить меру подобия онтологий, а также идентифицировать сходные концепции.
3. Разработать метод объединения онтологий, позволяющий построить общую онтологию, включающую все концепции исходных онтологий и дополненную аксиомами эквивалентности сходных концепций исходных онтолгий.
4. Реализовать разработанные методы в рамках инструментария сравнения и объединения онтологий.
5. Провести экспериментальную проверку разработанного инструментария на онтологиях различного размера, относящихся к различным предметным областям.

Схема инструментария сравнения и объединения онтологий, который планируется реализовать в данной работе, представлена на рисунке 1.

На вход подаются сраниваемые онтологии, а на выходе получается объединенная онтология и мера соотвествия. Дополнительной опциональной возможностью может являться предпросмотр и корректировка результата сравнения пользователем.

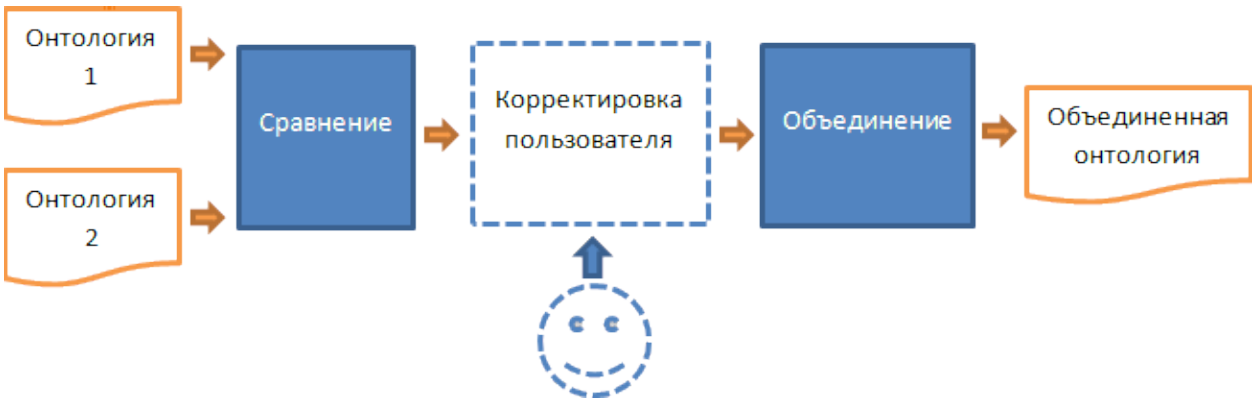

Рисунок 1: Схема работы инструмента сравнения и объединения онтологий

Обзор предметной области

Онтология представляет собой структуру данных, содержащую все релевантные классы объектов, их связи и правила (теоремы, ограничения), принятые в области, которую она описывает. Для идентификации классов используются унифицированные идентификаторы ресурсов (URI).

Для описания онтологий используются языки описания метаданных, наиболее употребительными из которых являются RDF и OWL.

Resource Description Framework (RDF) [15.] — это разработанная консорциумом Всемирной паутины[13.] модель для представления данных, в особенности — метаданных. RDF представляет утверждения о ресурсах в виде, пригодном для машинной обработки.

Ресурсом в RDF может быть любая сущность — как информационная (например, веб-сайт или изображение), так и неинформационная (например, человек, город или некое абстрактное понятие). Утверждение, высказываемое о ресурсе, имеет вид «субъект — предикат — объект» и называется триплетом. Утверждение «небо голубого цвета» в RDF-терминологии можно представить следующим образом: субъект — «небо», предикат — «имеет цвет», объект — «голубой». Для обозначения субъектов, предикатов и объектов в RDF используются URI. Множество RDF-утверждений образует ориентированный граф, в котором вершинами являются субъекты и объекты, а рёбра помечены предикатами.

Web Ontology Language (OWL) [14.] — язык описания онтологий для семантической паутины. Язык OWL позволяет описывать классы и отношения между ними, присущие для веб-документов и приложений. Рекомендован консорциумом Всемирной паутины. Имея в основе модель RDF, OWL добавляет к ней мощный механизм аксиом и ограничений.

URI-сравнение

Т.к. сущности в онтологии однозначно идентифицируется URI, самым очевидным подходом сравнения классов, используемых в разных онтологиях, является сравнение их URI. Классы с совпадающими URI помечаются эквивалентными (owl:equivalentClass), остальные несовпадающими.

Примером использования данного подхода может служить средство сравнения онтологий Bubastis[6.], разработанное в Европейском институте биоинформатики. Получив на вход две онтологии, Bubastis рассматривает все аксиомы относительно классов (любой RDF-трипплет считается аксиомой. Объявление класса (вида pl:Java a rdfs:Class) тоже считается аксиомой) и находит, какие аксиомы были добавлены, удалены или изменены. Под совпадение аксиом понимается точное совпадение всех используемых в них сущностей и их URI.

Пример вывода инструмента Bubastis на двух версиях одной и той же онтологии представлен на русунке 2.

```
---
!New class!: URI http://www.ebi.ac.uk/efo/EFO_0001561
label: kainic acid
+ SubClassOf('kainic acid' 'chemical compound')
+ SubClassOf('kainic acid' 'drug')
+ SubClassOf('kainic acid' ObjectUnionOf(ObjectSomeValueFrom('has_role' 'antinem
atodal drug') ObjectSomeValueFrom('has_role' 'excitatory amino acid agonist')))

---
!New class!: URI http://www.ebi.ac.uk/efo/EFO_0001500
label: carbon monoxide
+ SubClassOf('carbon monoxide' 'chemical compound')
+ SubClassOf('carbon monoxide' ObjectSomeValueFrom('has_role' 'mitochondrial res
piratory-chain inhibitor'))

---
!New class!: URI http://www.ebi.ac.uk/efo/EFO_0001569
label: mesalamine
+ SubClassOf('mesalamine' 'chemical compound')
```

Рисунок 2: Пример вывода инструмента Bubastis

Другим примером инструмента сравнения, основанного на сопоставлении URI, является OWLDiff [16.], разрабатываемый в Чешском техническом университете. Он также, как и Bubastis, сравнивает наборы аксиом

онтологий, считая аксиомы совпадающими, если совпадают все используемые в них URI. В отличие от инструмента Bubastis, у OWLDiff есть графический интерфейс, рисующий два иерархических дерева сравниваемых онтологий и расскрашивающий классы и аксиомы в нем двумя цветами: зеленым для несовпадающих и синим для совпадающих, как это показано на рисунке 3.

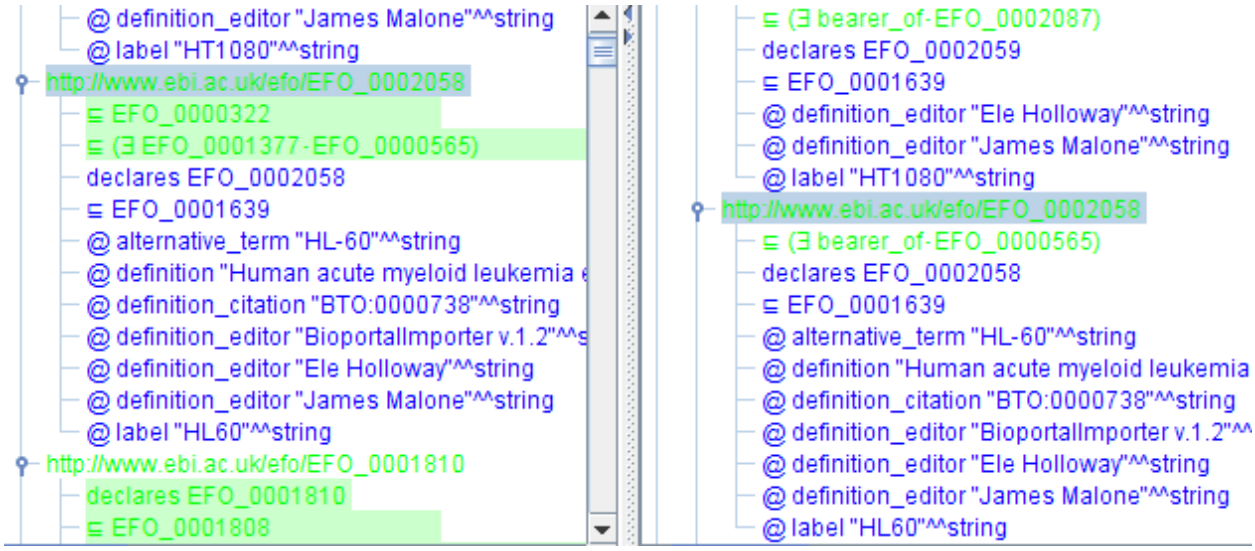

Рисунок 3: Пример работы инструмента OWLDiff

Данный подход работает быстро и хорош для сравнения различных версий одной и той же онтологии. Однако, для разных онтологий он практически неприменим вследствие неуникального именования[1.]: одной и той же сущности в разных онтологиях могут соотвествовать различные URI, что сделает все аксиомы с использованием этих сущностей несовпадающими с точки зрения инструментов Bubastis и OWLDiff.

Лексикографическое сравнение

Из неуникальности именования логично следует лексикографический подход к сравнению: классы считаются эквивалентными, если их имена совпадают.

Такой подход решает проблему различных URI для одних и тех же сущностей, используемых в разных онтологиях, однако и у него имеется ряд очевидных недостатков: омонимы (например, язык программирования java и java-кофе) будут сочтены подобными, в то время как синонимы и различные написания одних и тех же слов (американское tumor и британское tumour) окажутся несовпадающими.

Способ решения проблемы различного написания предлагается, например, в работе A. Maedche[5.]: вводится понятие расстояния между словами, измеряемого отношением минимального количества замен, вставок и удалений букв, требуемого для получения из одного слова другого, к общему числу букв в слове. Например, для получения пятибуквенного tumor из tumour потребуется одно удаление, следовательно, расстояние оказывается равным 1/5. Введя нижний порог расстояния между названиями сущностей и считая названия, малоудаленные друг от друга, совпадающими, можно частично решить проблему различных написаний.

Однако, этот подход не решает проблемы нахождения синонимов и омонимов, а также вводит дополнительный «шум» за счет ошибочно сочтенных за эквивалентные похожих в написании, но различных по значению слов (ontology и anthology).

Сравнение с учетом значений

Решение проблемы различения омонимов и сопоставления синонимов было предложено в работе James Z. Wang and Farha Ali [11.] за счет сопоставления именам классов синсетов базы данных слов английского языка WordNet [7.]. Все слова в WordNet подразделены на “синсеты” (synsets) — множества синонимов, имеющих общий смысл. Между синсетами существуют связи гипероним-гипоним (более общее – более узкое понятие) и холоним-мероним (целое-часть).

Для выбора правильного синсета, соотвествующего названию данного класса, (например для слова «java» будет найдено несколько синсетов: «язык программирования», «напиток из кофейных зерен» и др.) рассматриваются отношения между классами в онтологии и синсетами в базе WordNet. Синсет сопоставляется классу, если среди классов, связанных с рассматриваемым иерархическим отношением или отношением owl:partOf-owl:hasPart, найдется класс, которому соотвествует синсет, связанный с рассматриваемым синсетом исходного класса аналогичным отношением в базе WordNet (иерархическим отношениям (rdfs:subClassOf) в онтологии сопоставляются отношения гиперним-гипоним в WordNet, а отношениям owl:partOf-owl:hasPart мероним-холоним).

Классы считаются подобными, если их синсеты совпадают.

Мера сходства онтологий вычисляется как отношение количества пар подобных классов к общему числу классов в обеих онтологиях(с точностью до подобия).

Такой подход решает проблему эквивалентности синонимов, но обладает большим недостатком: он ограничен словами базы WordNet и не сможет сопоставить классы, чьи названия в ней не содержатся. Алгоритм также не решает проблему различных написаний (например, термин object oriented programming language в базе WordNet найден не будет, в отличие от object-oriented [через дефис] programming language). Алгоритм также никак не учитывает при оценке сходства онтологий описанные в них совйства и отношения между классами.

Онтология верхнего уровня

Еще одним подходом к сравнению онтологий является создание так называемой “онтологии верхнего уровня” [19.] – онтологии, описывающей общие понятия, единые для всех доменов.

Если бы такая онтология существовала, то сравнение существующих в ней классов двух онтологий свелось бы к решению вопроса о существовании в онтологии верхнего уровня аксиомы эквивалентности этих классов.

Однако, очевидно, что никакая онтология не может включать все понятия всех возможных онтологий и поэтому онтология верхнего уровня не решает проблему сравнения онтологий.

Работа в направлении создания онтологий верхнего уровня ведется, примером может служить онтология SUMO [8.].

Следует отметить, что подход к сравнению онтологий с использованием онтологии верхнего уровня хорошо сочетается с подходом сравнения с учетом значений: вместо сопоставления классам синсетов базы слов английского языка WordNet можно сопоставлять им классы онтологии верхнего уровня [12.].

Обобщение

Задача сравнения онтологий делится на два класса: сравнение разных версий одной и той же онтологии и сравнение двух произвольных онтологий.

Первая задача легко решается сравнением URI и нахождением добавленных и удаленных аксиом (перебором). Но этот метод практически не дает результатов при сравнении произвольных онтологий вследствие неуникальности именования.

Для сравнения произвольных онтологий используется лексикографический подход, но и он подвержен недостаткам, в частности, он не отличает омонимы и не видит связи между синонимами.

Метод сравнения с учетом значений, предложенный James Z. Wang and Fahra Ali, помогает различать омонимы и сопоставлять синонимы, но справлятся с этой задачей только в рамках используемой им онтологии верхнего уровня

(или базы слов WordNet), он также не учитывает при вычислении меры сходства онтологий определенные в них свойства и аксиомы.

Таким образом, каждый из рассмотренных методов хорошо справляется с задачей сравнения онтологий лишь на онтологиях определенного типа и не дает хороших результатов на онтологиях, к этому типу не относящихся. Для успешного сравнения произвольных онтологий следует попытаться объединить достоинства всех рассмотренных выше методов.

Методы сравнения и объединения онтологий

Метод сравнения онтологий

Предлагаемый в данной работе метод сравнения онтологий сочетает в себе три подхода: URI-сравнение, анализ значений слов и лексикографический анализ.

В методе также учитываются аксиомы, накладываемые на рассматриваемые классы каждой из онтологий. К таким аксиомам относятся иерархические отношения, отношения часть-целое, а также отношения, индуцированные присутсвующими в онтологиях наборами свойств и ограничений.

Схема метода сравнения представлена на рисунке 4.

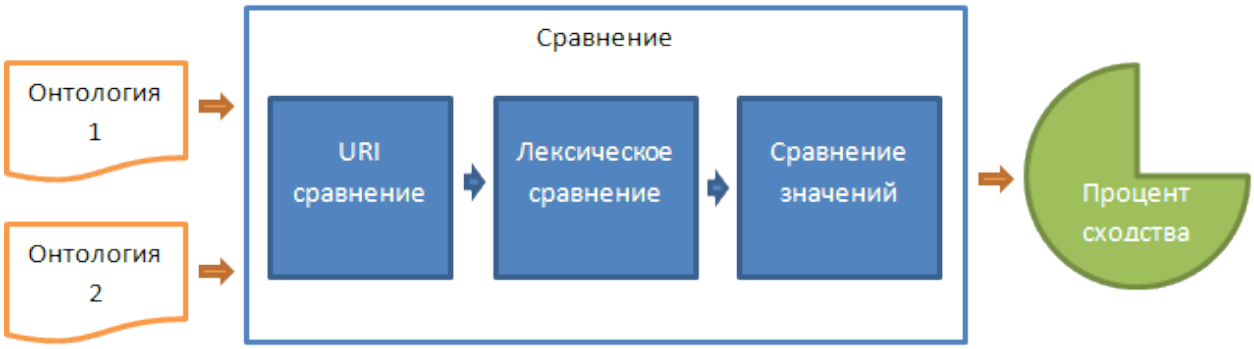

Рисунок 4: Общая схема сравнения онтологий

URI-сравнение

Под URI-сравнением понимается непосредственное сравненит URI рассматриваемых классов.

Лексикографическое сравнение

При лексикографическом сравнении сопоставляются названия классов. Под названием класса понимается значение его свойства rdfs:label[1], куда согласно W3C[13.] спецификации принято помещать информацию о названии. Однако,

1 http://www.w3.org/TR/rdf-schema/#ch_label

это требование не всегда соблюдается. И в случае, когда свойство rdfs:label для рассматриваемого класса отсуствует в онтологии, в качестве название класса берется фрагмент URI.

Еще одна трудность, возникающая при лексикографическом сравнении сопряжена с различным написанием одних и тех же слов. Например, object-oriented programming language и object oriented [без дефиса] programming language. В работе Maedche[5.] рассматривается алгоритм, решающий эту проблему за счет вычисления расстояния между словами и введения минимального порога расстояния. Слова, удаленные на величину, не превосходящую порог, считаются лексикографически совпадающими, а остальные – различными. Расстояние вычисляется как отношение минимального количества вставок/удалений/замен букв, необходимого для получения одного слова из другого, к общему числу букв в слове. Однако такое решение добавило бы лишнего «шума» из несовпадающих по смыслу, но малоотличающихся по написанию слов (ontology и anthology). Вместо этого было решено ограничиться легким преобразованием слов к стандартному виду: переводу в нижний регистр, удалению лишних пробелов и замены нижних подчеркиваний и дефисов на пробелы. Таким образом, слова наподобие object-oriented programming language и object oriented [без дефиса] programming language приводятся к общему виду, но более существенные изменения (добавление/удаение/замена букв) не происходят.

Сравнение значений

Сравнение значений названий классов осуществляется при помощи базы данных слов английского языка WordNet [7.], разрабатываемой в Принстонском университете под руководством пофессора психологии Джорджа А. Миллера.

В базе WordNet хранятся прилагательные, существительные, глаголы и наречия английского языка, сгруппированные в «синсеты» – множества

синонимов, имеющих общий смысл, и список кратких общих определений и глоссов (применений в реальных текстах). Также в базе хранятся различные семантические связи между синсетами, такие как связь холоним/мероним (общее/часть, например «скелет» и «позвоночник»), гипероним/гипоним (более общее понятие/более частное, например «напиток» и «кофе»).

Базу данных WordNet можно свободно использовать в коммерческих и научных целях. Для работы с ней существует несколько программ, множество интерфейсов (в том числе, веб-интерфейс) и API, реализуемых на большинстве возможных языков программирования.

Для сравнения значений названий классов с помощью базы данных WordNet, для каждого из названий предварительно ищется «синсет» в базе. Отображение классов онтологии на синсеты базы WordNet происходит следующим образом. В базе WordNet находятся все синсеты, в которых присутствует название класса. Например, для названия «java» найдется три синсета, как показано на рисунке 5: «остров в Индонезии», «напиток, приготовленных из кофейных зерен» и «платформо-независимый объектно-ориентированный язык программирования».

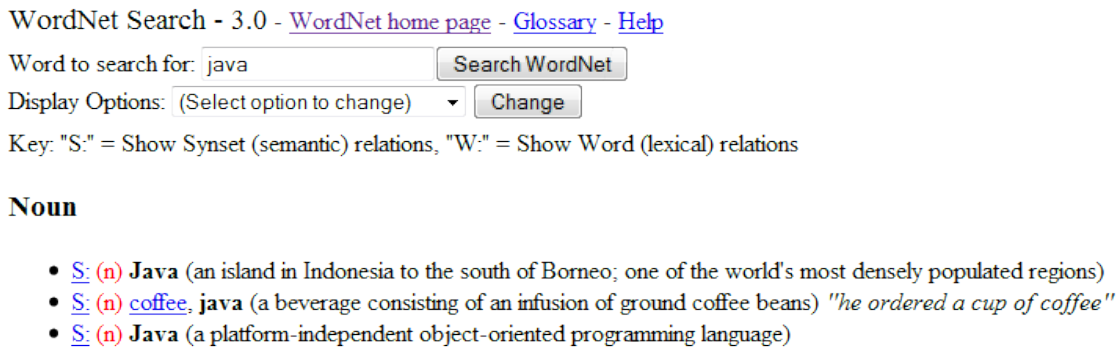

Рисунок 5: Синсеты, нацденный в базе WordNet для слова "java"

Чтобы выбрать из найденных синсетов правильный, иерархические отношения (rdfs:subClassOf), и отношения частное/целое (owl:partOf/owl:hasPart) для класса в онтологии сравниваются соотвественно с отношениями гиперним/гипоним и холоним/мероним для синсета в базе WordNet. Каждая такая пара отношений считается подобной, если классам онтологии, связанным первым отношением пары, соотвествуют синсеты, связанные вторым. Примером пары подобных отношений может служить отношение «ontopl:java rdfs:subClassOf ontopl:object-oriented programming language» онтологии OntoPL, представленной на рисунке 6, и отношение гиперним/гипоним между синсетами «платформо-независимый объектно-ориентированный язык программирования» и «язык программирования, поддерживающий понятие объектов, их свойств и методов обработки».

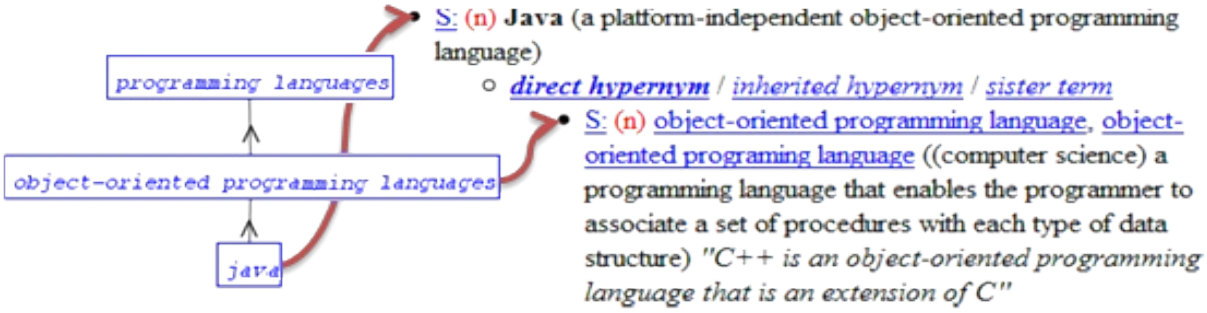

Рисунок 6: Сопоставление отношений в онтологии отношениям в базе WordNet

Мера соотвествия синсета классу может быть вычеслена по формуле (1), параметризованной весами отношения общее/частное, и отношения целое/часть.

$$Sim(c,s)=\frac{\omega^{(h)}*R^{(h)}(c,s)}{min(R^{(h)}(c),R^{(h)}(s))}+\frac{\omega^{(m)}*R^{(m)}(c,s)}{min(R^{(m)}(c),R^{(m)}(s))}$$, где

$R^{(h)}(c, s)$ – число подобных отношений общее/частное для класса и синсета,

$R^{(h)}(c)$ – число иерархических отношений, в которых состоит класс в онтологиии,

$R^{(h)}(s)$ – число отношений гиперним/гипоним, в которых состоит синсет,

$R^{(p)}(c, s)$ – число подобных отношений целое-часть для класса и синсета,

$R^{(p)}(c)$ – число отношений owl:partOf/owl:hasPart, в которых состоит класс в онтологиии,

$R^{(p)}(s)$ – число отношений холоним/мероним, в которых состоит синсет,

$\omega^{(h)} + \omega^{(m)} = 1$,

$0 \leq \omega^{(h)}, \omega^{(m)} \leq 1$,

$\omega^{(h)}$ – вес отношения общее/частное ,

$\omega^{(m)}$ – вес отношения целое/часть.

Выражения вида 0 / 0, соотвествующие ситуации, когда отношения какого-либо из вышеперечисленных типов для класса или синсета отсутствуют, будем считать равными 0 (тем самым, значение (синсет), не подтвержденное ни одним отношением, рассматриваться не будет).

Мера сходства

После того, как описаны механизмы, используемые при сравнении классов, можно ввести меру сходства, получаемую после такого сравнения.

Мера сходства двух классов вычисляется по формуле (2), параметризованной весами лексикографического и сходства с учетом значений названий.

$$Sim(c^{(1)}, c^{(2)}) = \frac{(1-\varepsilon) * f(\omega^{(WN)}, \omega^{(L)}) * n}{max(N^{(1)}, N^{(2)})} + \varepsilon * f(\omega^{(WN)}). \quad (2)$$

, где

$0 \leq f(\omega^{(URI)}, \omega^{(WN)}, \omega^{(L)}) \leq 1$,

$\omega^{(WN)}$ – вес сходства с учетом значений,

$\omega^{(L)}$ – вес лексикографического сходства,

n – число эквивалентных аксиом относительно рассматриваемых классов,

$N^{(i)}$ – общее число аксиом относительно класса $C^{(i)}$ в онтологии $O^{(i)}$,

ε – малая константа (параметр),

Функция $f(\omega^{(WN)}, \omega^{(L)})$ вычисляется следующим образом:

1. Если URI классов совпадают (то не нужно рассматривать синсеты и лексикографические связи, т.к. URI уникально идентифицируют сущность), то $f(\omega^{(WN)}, \omega^{(L)}) = 1$
2. Иначе, если синсеты для обоих классов найдены и
    1. синсеты не совпадают (разные значение), то $f(\omega^{(WN)}, \omega^{(L)}) = 0$
    2. синсеты совпадают (значение одинаковое и рассматривать лексикографическое сходство смысла нет, т.к. оно уже было бы учтено при поиске синсета), то $f(\omega^{(WN)}, \omega^{(L)}) = \omega^{(WN)}$
3. Иначе, если хотя бы для одного из классов синсет не найден, но лексикографически названия классов совпадают, то $f(\omega^{(WN)}, \omega^{(L)}) = \omega^{(L)}$
4. Иначе $f(\omega^{(WN)}, \omega^{(L)}) = 0$.

За счет домножения на отношение числа совпадающих аксиом к общему числу аксиом, учитываются связи и не окажутся подобными классы-омонимы (java-кофе и java-язык программирования), не найденные в базе WordNet.

Функция $f(\omega^{(WN)})$ вычисляется аналогично функции $f(\omega^{(WN)}, \omega^{(L)})$ (пп. 1-2, 4), но не учитывает лексикографические совпадения. Она и малая константа ε вводятся для того, чтобы в онтологиях, содержащих классы с совпадающими URI или синсетами, но без пересечения по аксиомам (например, если в одной из онтологий прописаны только owl:partOf отношения, а в другой исключительно иерархические связи) для этих классов, такие классы все же оказались помеченными подобными.

Заметим, что как и требуется, $0 \leq Sim(c^{(1)}, c^{(2)}) \leq 1$ и для классов с совпадающими URI и полностью совпадающим набором аксиом (например, если сравнивать класс с самим собой) $Sim(c, c) = 1$.

Мера сходства двух онтологий, рассматриваемая в данной работе, представлена в формуле (3) и является отношения суммарного сходства пар классов этих онтологий к общему числу различных классов (с точностью до вышенайденного подобия: классы $c^{(1)}$ и $c^{(2)}$ подобны, если $Sim(c^{(1)}, c^{(2)}) > 0$).

$$Sim(O^{(1)}, O^{(2)}) = \frac{\Sigma\, sim(c^{(1)}, c^{(2)})}{|O^{(1)} \cup O^{(2)}|} \qquad (3)$$

Метод объединения онтологий

Неотъемлемой частью работы с онтологией является использование машин вывода [10.] – программ, которая выполняющих логический вывод из

предварительно построенной базы фактов и правил в соответствии с законами формальной логики.

Например, из наличия в онтологии аксиомы эквивалентности классов A и B (A owl:equivalentClass B) может быть выведено, что все утверждения, справедливые для класса A, справедливы также для класса B и наоборот.

Таким образом, после того, как на этапе сравнения в онтологиях были найдены пары подобных классов (для которых $Sim(c^{(1)}, c^{(2)}) > \Theta$, где Θ – пороговое значение, являющееся параметром), для объединения онтологий в общую путем достаточно записать в нее все аксиомы базовых онтологий, дополнив их аксиомами эквивалентности (owl:equivalentClass) для найденных пар подобных классов. Запись каких-либо еще аксиом не требуется, т.к. все они могут быть получены из объединенной онтологии при помощи машины вывода.

Вышеописанный алгоритм осуществляет автоматическое объединение онтологий, но оно может быть также проведено в не полностью автоматическом режиме, когда перед объединением в общую онтологию, подобные классы, найденные при сравнении, показываются пользователю, и эквивалентными в объединенной онтологии помечаются лишь те из них, которые были одобрены пользователем.

Реализация

Для проверки вышеизложенного метода был реализован прототип инструмента сравнения и объединения онтологий. В качестве языка программирования был выбран язык Java, в силу его мультиплатформенности, а также наличия библиотек для работы с онтологиями и базой WordNet. Для преобразования файлов в форматах OWL, OBO и RDF (N3, XML), хранящих онтологии, в объектно-ориентированное представление использовалась библиотека OWL API[4.], разрабатываемая в университете Манчестера. Интеграция с базой данных слов английского языка WordNet осуществлялась при помощи библиотеки Java API for WordNet Searching (JAWS)[17.], разрабатываемой Б. Спелл в Южном методистском университете.

Общая схема

Общая схема работы прототипа представлена на рисунке 7.

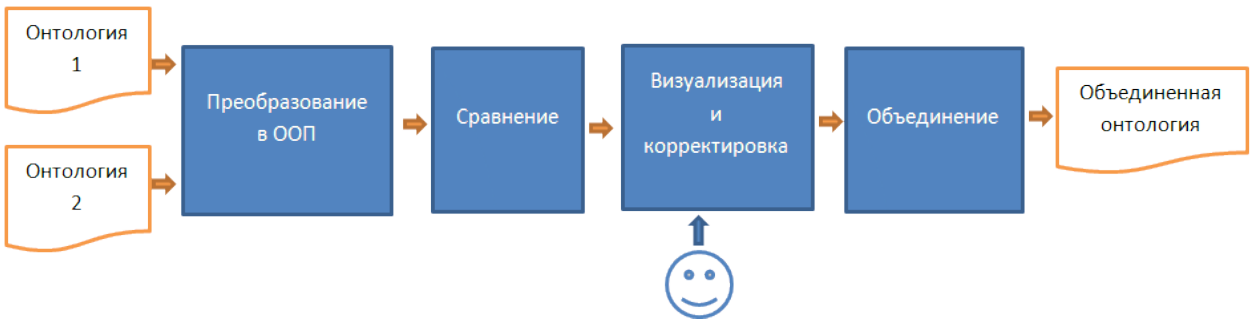

Рисунок 7: Общая схема работы инструмента сравнения и объединения онтологий

Сравнение онтологий делится на 4 шага: преобразование файла онтологии в объектно-ориентированное представление, сравнение онтологий, визуализация результатов сравнения для ознакомления с ними пользователя и, с учетом корректировок, внесенных пользователем, сохранение обобщенной онтологии.

Рассмотрим эти шаги более детально.

Преобразование файла онтологии в объектно-ориентированное представление

На вход подаются две сравниваемые онтологии в формате OWL, RDF (XML, N3) или OBO. С помощью библиотеки OWL API они преобразуются во внутреннее объектно-ориентированное представление, хранящее информацию о классах, свойствах, отношениях и ограничениях исходной онтологии. Для каждого класса онтологии хранится его название, URI, а также отношения, в которых он находится с другими классами. За преобразование отвечает класс OntologyGraphBuilder, работа которого представлена на рисунке 8.

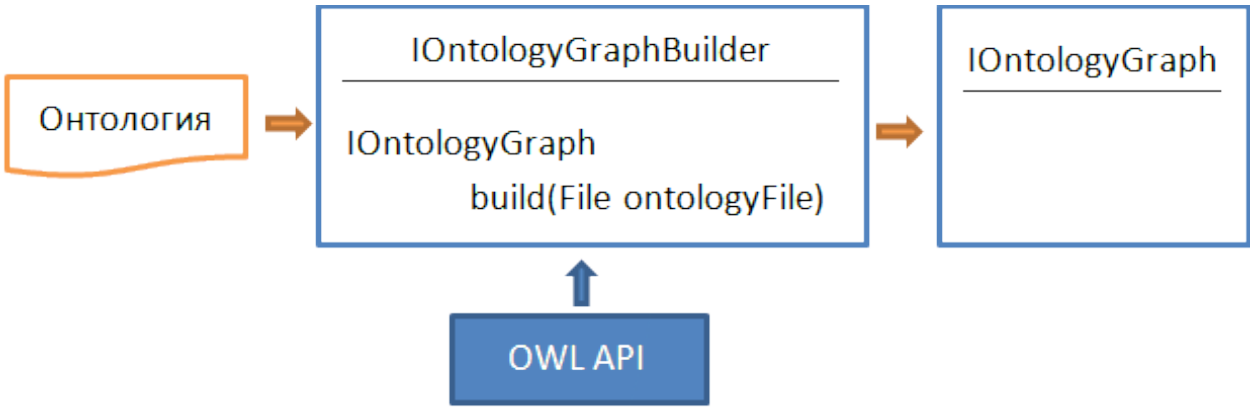

Рисунок 8: Преобразование файла онтологии в объектно-ориентированное представление

На этом же этапе названия классов нормализуются (дефисы и подчеркивания заменяются на пробелы и все слова переводятся в нижний регистр), что будет использовано на этапе лексического сравнения для решения проблемы незначительно отличающихся написаний.

Сравнение

Сравнение состоит из трех шагов: URI-сравнения, сравнения с учетом значений и лексикографического сравнения. Схема сравнения представлена на рисунке 8.

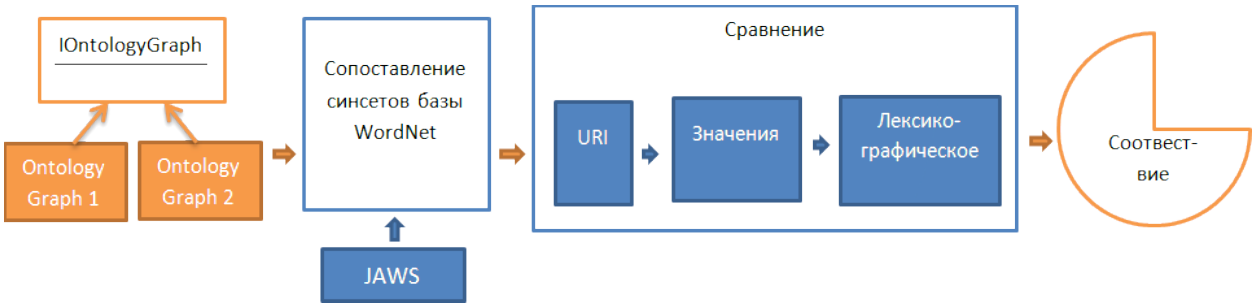

Рисунок 9: Схема сравнения онтологий

Наиболее простым алгоритмом сравнение был бы полный перебор всех пар классов онтологий и подсчет мер их подобия по формуле (2). Однако такой подход был бы по меньшей мере квадратичен (если для каждого класса число аксиом ограничено и не очень велико) по времени относительно числа классов онтологий. Для того, чтобы оптимизировать процесс, на шаге преобразования файла онтологии в объектно-ориентированное представление осуществляется индексирование классов по URI и названиям (за счет помещения их в хэш-таблицы, ключами которых являются соотвественно URI и названия классов). При отображении классов на синсеты базы WordNet, осуществляемом непосредственно перед сравнением, строится еще одна хэш-таблица, ключами которой являются синсеты. После такой предобработки само сравнение происходит линейно: осуществляется проход по классам одной из онтологий и каждый из них сравнивается только с классами с совпадающими URI, синсетами или названиями.

За сопоставление классов онтологий отвечает класс OntologyConceptMapper, параметризуемый набором реализаций интерфейса IComparator. Каждый из наследников IComparator реализует стратегию сравнения сущностей. На

данный момент используется три вида стратегий: UriComparator, SynsetComparator и LexicalComparator для сравнения с учетом URI, значений и лексикографически соотвественно. Однако в будущем этот набор может быть расширен.

Процесс сопоставления классам онтологий сиснсетов базы WordNet выполняется аналогично, с использованием класса SynsetMapper, параметризованного единственной стратегией ConceptToSynsetComparator.

Оба класса OntologyConceptMapper и SynsetMapper реализуют общий интерфейс IMapper, отвечающий за отображение одного набора сущностей на другой набор.

Схема иерархии классов, наследующих IMapper представлена на рисунке 10.

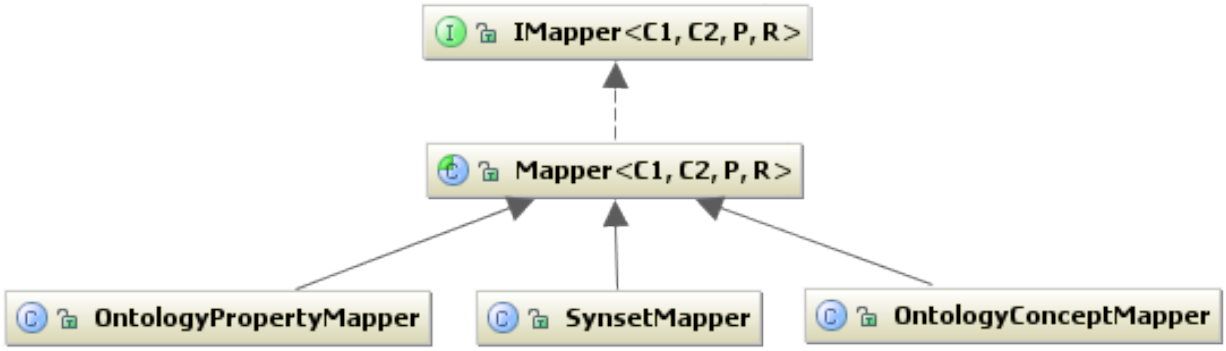

Рисунок 10: Иерархия классов

Визуализация и корректировка

Результат сравнения онтологий визуализируется в виде графа с вершинами-классами и ребрами-отношениями. Найденные эквивалентные классы объединяются дополнительными внешними вершинами, при выделении которых можно просмотреть информацию о причинах эквивалентности (как показано на рисунке 11).

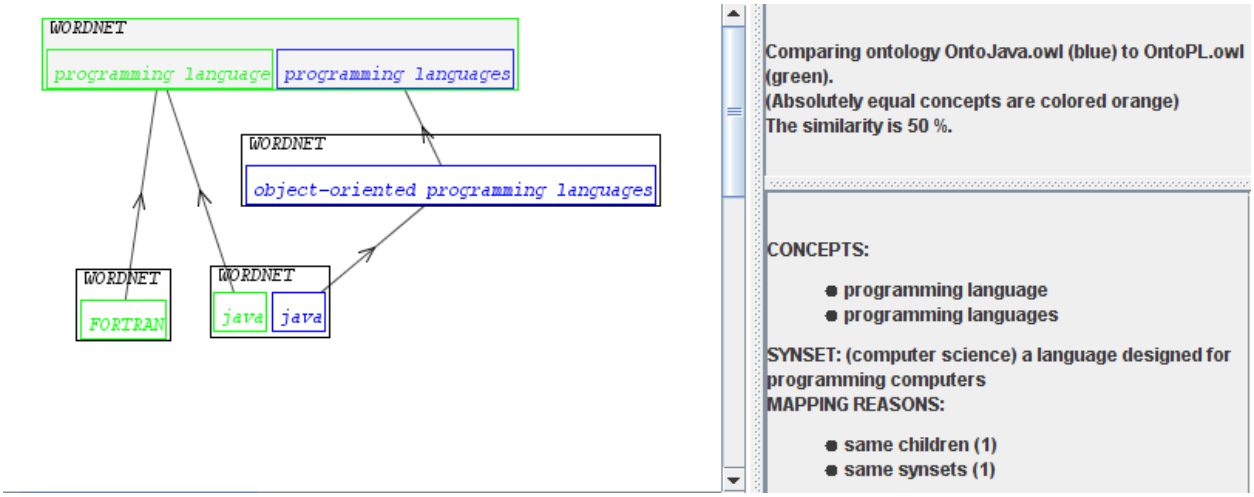

Рисунок 11: Визуализация

Пользователь может удалить неверно найденное на его взгляд соотношение эквивалентности, используя контекстное меню, появляющееся при щелчке мыши на соотвествующей паре помеченных эквивалентными классов, как это показано на рисунке 12.

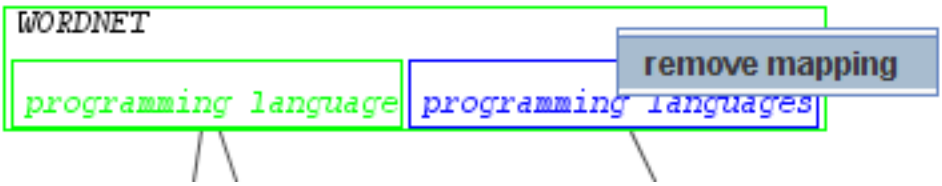

Рисунок 12: Корректировка

Сохранение

После того, как пользователь ознакомился с результатом сравнения и внес необходимые поправки, он может сохранить объединенную онтологию в формате OWL.

При сохранении в объединенную онтологию при помощи библиотеки OWL API запишутся аксиомы исходных онтологий а также найденные в процессе сравнения и одобренные пользователем аксиомы эквивалентности между классами.

Эксперименты

Чтобы проиллюстрировать эффективность предлагаемого подхода к сравнению и объединению онтологий, был проведен ряд экспериментов. Сравнивались как разные версии одной онтологиии, так и различные онтологии.

Эксперимент 1. Сравнение онтологии с самой собой

В качестве онтологии была взята представленная в приложении 1 онтология OntoJava.owl. Как и ожидалось, результат сравнение ее с самой собой выдал полное совпадение (рисунок 13).

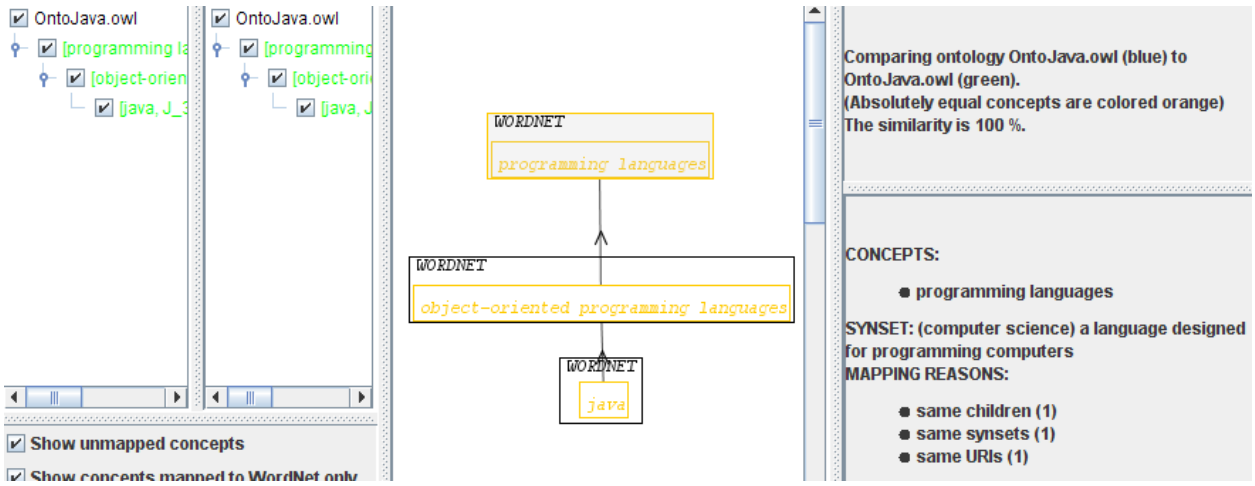

Рисунок 13: Сравнения онтологии OntoJava с самой собой

Для сравнения иструмент Bubastis на данной онтологии выдал тот же результат (рисунок 14).

```
Bubastis version 1.1
attempting to load file:C:/Users/sony/Desktop/OntoJava.owl
loading ontology 1 complete
attempting to load file:C:/Users/sony/Desktop/OntoJava.owl
loading ontology 2 complete
===Output is to console===

No changes between ontologies
Bubastis Report Complete
```

Рисунок 14: Результат сравнения онтологии OntoJava с собой с помощью инструмента Bubastis

Эксперимент 2. Онтологии, содержащие классы с разными URI, но одинаковые по смыслу

Рассматриваемые в примере онтологии OntoPLFull.owl и OntoJavaCSharp.owl представлены в приложении 2. Результат их сравнения показан на рисунке 15. (Классы онтологии OntoPLFull представлены зеленым цветом, а классы OntoJavaCSharp синим.)

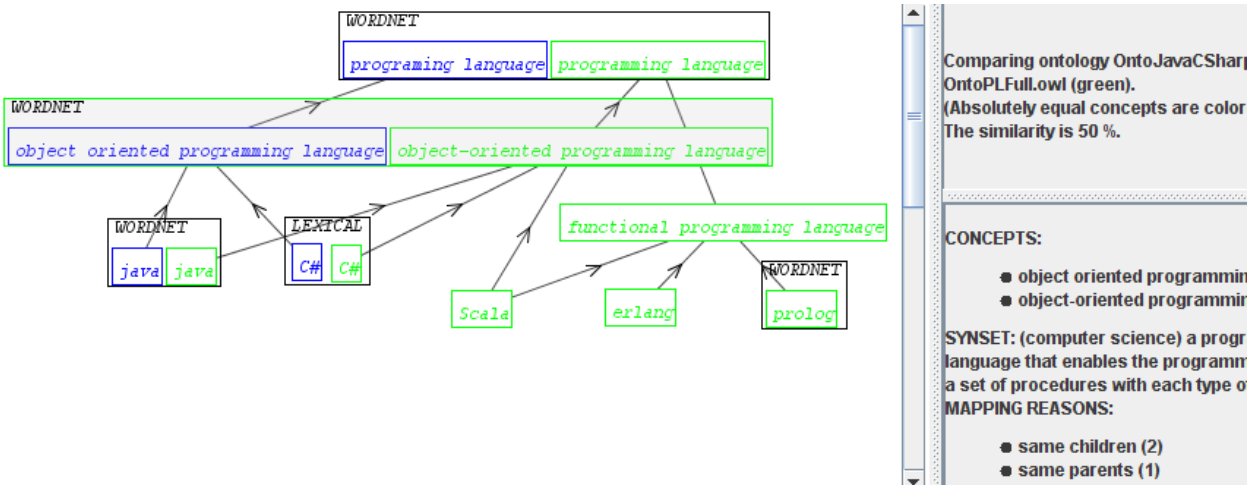

Рисунок 15: Сравнение онтологий OntoPLFull и OntoJavaCSharp

Как видно из рисунка инструмент справился с сопоставлением классов с разными написаниями названий (programming language и programing [с одной m] language), найдя оба написания в базе WordNet. За счет нормализации названий название object oriented (без дефиса) programming language так же было сопоставлено названию object-oriented programming language.

Иструмент так же смог сопоставить не найденные в базе WordNet, но лексикографически совпадающие и имеющие совпадающих предков классы C# и C#. Классу prolog онтологии OntoPLFull.owl не было найдено эквивалентного в онтологии OntoJavaCSharp.owl, но был сопоставлен синсет базы WordNet “a computer language designed in Europe to support natural language processing”.

В отличие от данного иснструмента, средства сравнения онтологий Bubastis и OWLDiff не нашли общего в рассматриваемых онтологиях (рисунок 16).

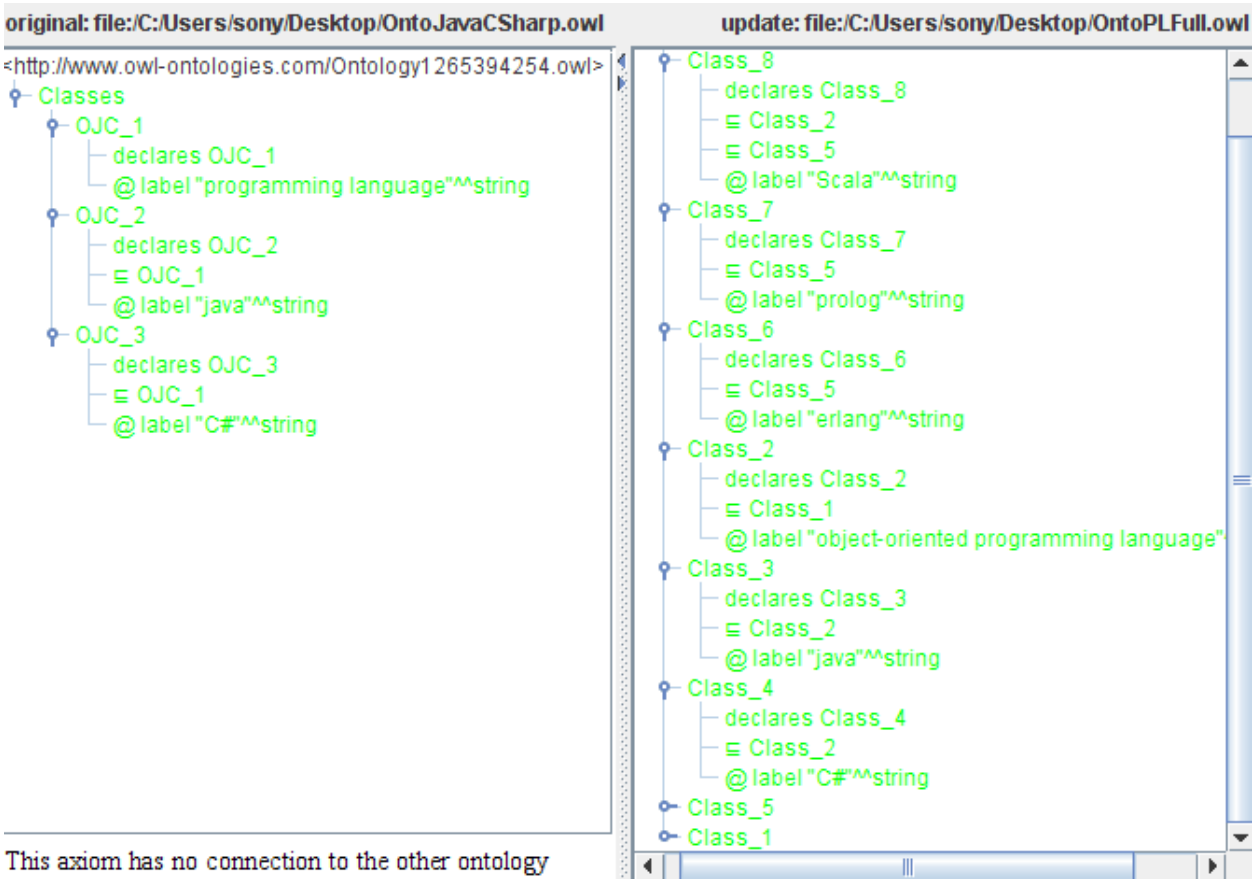

Рисунок 16: Сравнение онтологий OntoPLFull и OntoJavaCSharp с помощью инструмента OWLDiff (несовпадающие аксиомы и классы помечены зеленым)

Эксперимент 3. Сравнение онтологий, содержащих классы с одинаковыми названиями, но разные по смыслу

Рассматриваемые в примере онтологии OntoJava.owl и OntoDrink.owl представленные в приложениях 1, 3. Результат их сравнения показан на рисунке 17.

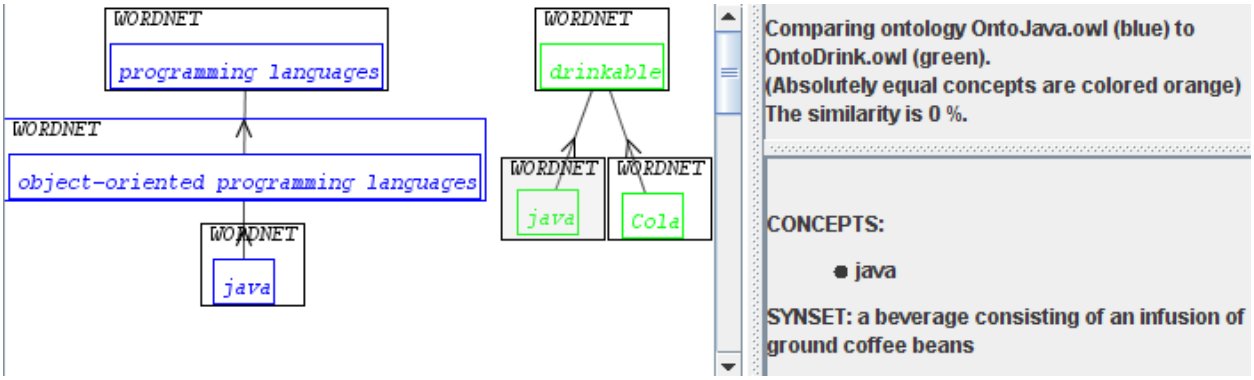

Рисунок 17: Сравнение онтологий OntoJava и OntoDrink

В обеих онтологих присутсвует класс с названием java, однако в случае онтологии OntoJava.owl это язык программирования, а в случае OntoDrink.owl – напиток. Инструмент справился с нахождением правильных синсетов в базе WordNet для каждого из классов и не счел их эквивалентными.

Средства OWLDiff и Bubastis также не нашли сходства.

Эксперимент 4. Сравнение разных версий одной и той же онтологии

Для сравнения были взяты две версии (94 и 114) онтологии Experimental Factor Ontology (EFO) [6.], содержащей более 2500 классов. Онтология EFO заимствует некоторые классы из других онтологий. В более ранних версия этим классам сопоставлялся новый EFO-URI (вида, http://www.ebi.ac.uk/efo/EFO_...), в то время как в более поздних было решено использовать оригинальный URI (например, http://www.ifomis.org/bfo/1.1/snap#MaterialEntity для класса взятого из

онтологии BFO[2.]). Версия 94 относится к более ранним, в то время как 114 к более поздним. Из-за этого результат сравнения с помощью представленного в данной работе инструмента оказался более точным, чем результат, выдаваемый инструментами сравнения, основывающимися только на совпадении URI. На рисунке 18 классы с совпадающими URI объединены в одну вершину и нарисованы оранжевым цветом, в то время как эквивалентные классы с разными URI окрашены в цвета соотвествующих онтологий (зеленый и синий) и объединены внешней вершиной.

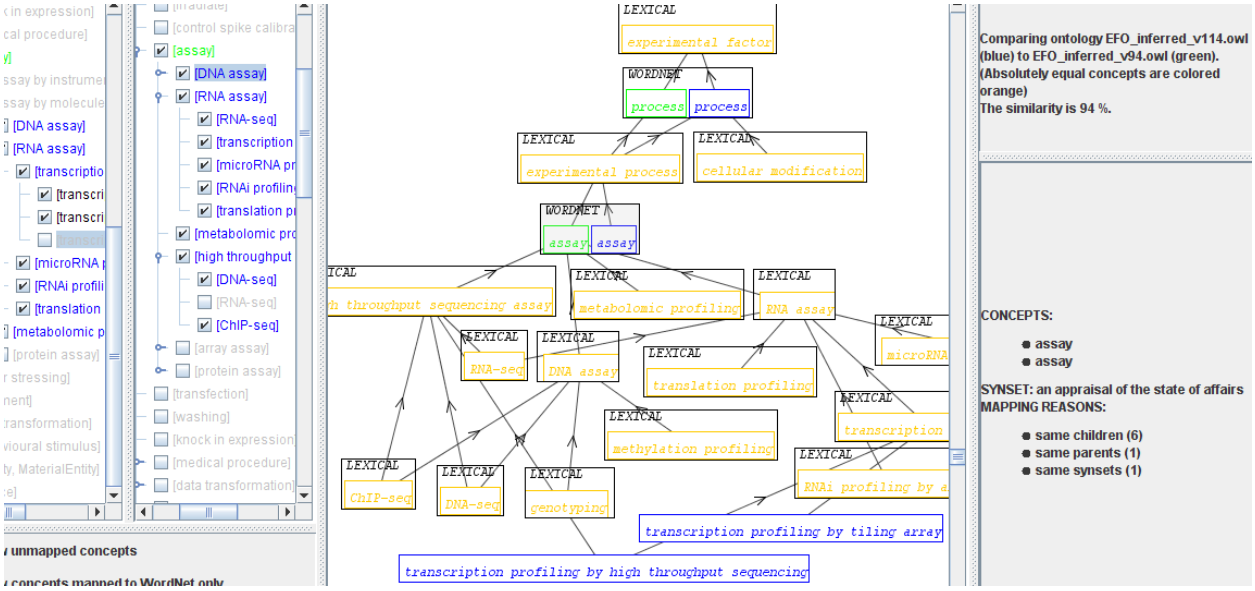

Рисунок 18: Сравнение разных версий онтологии EFO

Выводы и результаты

В рамках данной работы были разработан метод сравнения, позволяющий численно оценить подобие онтологий, а также идентифицировать сходные концепции. Разработан метод объединения онтологий, позволяющий построить общую онтологию, включающую все концепции исходных онтологий и дополненную аксиомами эквивалентности сходных концепций исходных онтологий.

Для практической проверки методов был реализован десктопный прототип инструмента сравнения и объединения онтологий, позволяющий сравнивать онтологии в форматах OWL, RDF(N3, XML) и OBO, визуализировать результаты сравнения и сохранять объединенную онтологию в формате OWL.

Для подтверждения эффективности предложенного метода была проведена серия экспериментов по сравнению онтологий различного размера и относящихся к разным предметным областям, показавшая, что метод успешно справляется с поставленной задачей, не допускает распространенных в существующих на данный момент инструментах ошибок: правильно сопоставляет синонимы и различные написания и различает омонимы.

Недостатком инструмента на данный момент являются большое потребление памяти для экономии времени, отсутствие сопоставления свойств, описанных с онтологиях, и невозможность находить значения для названий классов, не присуствующих в базе WordNet или не на английском языке.

Работа над устранение этих недостатков продолжается: сравнение свойств может быть выполнено методами, аналогичными сравнению классов (с учетом URI и лексикографической составляющих а так же аксиом, наложенных на свойства онтологиями).

Проблема с английским языком и сравнением онтологий, написанных на разных языкам открывает широкую перспективу для исследований и может, например, быть решена за счет умелой интеграции со словарями.

Ближайшим шагом развития инструмента является совместное с Европейским институтом биоинформатики[2] написание к нему веб-интерфейс и превращение его в веб-сервис по поиску сходных онтологий.

2 http://www.ebi.ac.uk/

Литература

1. Dean Allemang, James A. Hendler. Semantic web for the working ontologist: modeling in RDF, RDFS and OWL, Morgan Kaufmann 2008, ISBN: 978-0-12-373556-0

2. Robert Arp and Barry Smith Function, Role, and Disposition in Basic Formal Ontology, Bio-Ontologies 2008: Knowledge in Biology, 2008.

3. John Davies, Dieter Fensel, Frank van Harmelen, Towards the Semantic Web: Ontology-Driven Knowledge Management, John Wiley & Sons, ISBN 0-470-84867-7

4. Matthew Horridge, Sean Bechhofer. The OWL API: A Java API for Working with OWL 2 Ontologies. OWLED 2009, 6th OWL Experienced and Directions Workshop, Chantilly, Virginia, October 2009.

5. Alexander Maedche, Steffen Staab. Measuring Similarity between Ontologies Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web, Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management, EKAW02, volume 2473 of LNAI, Berlin, Springer Verlag, (2002)

6. Malone, J., Holloway, E., Adamusiak, T., Kapushesky, M., Zheng, J., Kolesnikov, N., Zhukova, A., Brazma, A., and Parkinson, H. (2010). Modeling sample variables with an Experimental Factor Ontology. Bioinformatics in press.

7. George A. Miller (1995). WordNet: A Lexical Database for English. Communications of the ACM Vol. 38, No. 11: 39-41.

8. Pease, A., Niles, I., and Li, J. 2002. The Suggested Upper Merged Ontology: A Large Ontology for the Semantic Web and its Applications. In Working Notes of the AAAI-2002 Workshop on Ontologies and the Semantic Web, Edmonton, Canada, July 28-August 1, 2002.

9. Russell, Stuart J.; Norvig, Peter (2003), Artificial Intelligence: A Modern Approach (2nd ed.), Upper Saddle River, New Jersey: Prentice Hall, ISBN 0-13-790395-2, http://aima.cs.berkeley.edu/ , chpt. 2

10. E. Sirin, B. Parsia, B. Cuenca Grau, A. Kalyanpur, and Y. Katz. Pellet: A Practical OWL-DL Reasoner.Technical report, University of Maryland Institute for Advanced Computer Studies (UMIACS), 2005.

11. James Z. Wang and Farha Ali. An Efficient Ontology Comparison Tool for Semantic Web Applications, IEEE/WIC/ACM International Conference on Web Intelligence (WI 2005), September 2005. (Acceptance rate: 18%, 59/328)

12. James Z. Wang, Farha Ali, Pradip Srimani. An Efficient Method to Measure the Semantic Similarity of Ontologies, In Proceedings of the 3rd International Conference on Grid and Pervasive Computing (GPC-08), LNCS 5036, pp. 447-458, 2008

13. World Wide Web Consortium (W3C) Home Page http://www.w3.org/

14. OWL Web Ontology Language Semantics and Abstract Syntax http://www.w3.org/TR/owl-semantics/

15. Resource Description Framework (RDF) Model and Syntax Specification http://www.w3.org/TR/PR-rdf-syntax/

16. OWLdiff Home Page http://krizik.felk.cvut.cz/km/owldiff/

17. Java API for WordNet Searching (JAWS) http://lyle.smu.edu/~tspell/jaws/index.html

18. Семантическая паутина http://ru.wikipedia.org/wiki/Семантическая_паутина

19. Upper Ontology http://en.wikipedia.org/wiki/Upper_ontology_(information_science)

Приложение 1. Онтологии OntoJava.owl, OntoPL.owl и онтология, полученная в результате их сравнения и объединения.

OntoJava.owl

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
    xmlns="http://www.owl-ontologies.com/OntologyPL.owl#"
    xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:swrl="http://www.w3.org/2003/11/swrl#"
    xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://www.owl-ontologies.com/OntologyPL.owl">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="J_1">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >programming languages</rdfs:label>
  </owl:Class>
  <owl:Class rdf:ID="J_3">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="J_2"/>
    </rdfs:subClassOf>
```

```
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >java</rdfs:label>
  </owl:Class>
  <owl:Class rdf:about="#J_2">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >object-oriented programming languages</rdfs:label>
    <rdfs:subClassOf rdf:resource="#J_1"/>
  </owl:Class>
</rdf:RDF>
<!-- Created with Protege (with OWL Plugin 3.4.1, Build 536)  http://protege.stanford.edu -->
```

OntoPL.owl

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns="http://www.owl-ontologies.com/Ontology1265393898.owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
    xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:swrl="http://www.w3.org/2003/11/swrl#"
    xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://www.owl-ontologies.com/Ontology1265393898.owl">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="PL_3">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >FORTRAN</rdfs:label>
```

```
    <rdfs:subClassOf>
      <owl:Class rdf:ID="PL_1"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:about="#PL_1">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >programming language</rdfs:label>
  </owl:Class>
  <owl:Class rdf:ID="PL_2">
    <rdfs:subClassOf rdf:resource="#PL_1"/>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >java</rdfs:label>
  </owl:Class>
</rdf:RDF>
<!-- Created with Protege (with OWL Plugin 3.4.1, Build 536)  http://protege.stanford.edu -->
```

Merged.owl

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
    <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY owl2xml "http://www.w3.org/2006/12/owl2-xml#" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
    <!ENTITY OntologyPL "http://www.owl-ontologies.com/OntologyPL.owl#" >
                                    <!ENTITY Ontology1265393898 "http://www.owl-
ontologies.com/Ontology1265393898.owl#" >
]>
<rdf:RDF xmlns="file:///C:/Users/sony/AppData/Local/Temp/ontology10529.owl#"
```

```
 xml:base="file:///C:/Users/sony/AppData/Local/Temp/ontology10529.owl"
 xmlns:Ontology1265393898="http://www.owl-ontologies.com/Ontology1265393898.owl#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
 xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:OntologyPL="http://www.owl-ontologies.com/OntologyPL.owl#">
<owl:Ontology rdf:about=""/>
<!--
///////////////////////////////////////////////////////////////////////////////////
//
// Classes
//
///////////////////////////////////////////////////////////////////////////////////
 -->
<!-- http://www.owl-ontologies.com/Ontology1265393898.owl#PL_1 -->
<owl:Class rdf:about="&Ontology1265393898;PL_1">
    <rdfs:label rdf:datatype="&xsd;string"
        >programming language</rdfs:label>
    <owl:equivalentClass rdf:resource="&OntologyPL;J_1"/>
</owl:Class>
<!-- http://www.owl-ontologies.com/Ontology1265393898.owl#PL_2 -->
<owl:Class rdf:about="&Ontology1265393898;PL_2">
    <rdfs:label rdf:datatype="&xsd;string">java</rdfs:label>
    <owl:equivalentClass rdf:resource="&OntologyPL;J_3"/>
    <rdfs:subClassOf rdf:resource="&Ontology1265393898;PL_1"/>
```

```
    </owl:Class>
    <!-- http://www.owl-ontologies.com/Ontology1265393898.owl#PL_3 -->
    <owl:Class rdf:about="&Ontology1265393898;PL_3">
        <rdfs:label rdf:datatype="&xsd;string">FORTRAN</rdfs:label>
        <rdfs:subClassOf rdf:resource="&Ontology1265393898;PL_1"/>
    </owl:Class>
    <!-- http://www.owl-ontologies.com/OntologyPL.owl#J_1 -->
    <owl:Class rdf:about="&OntologyPL;J_1">
        <rdfs:label rdf:datatype="&xsd;string"
            >programming languages</rdfs:label>
    </owl:Class>
    <!-- http://www.owl-ontologies.com/OntologyPL.owl#J_2 -->
    <owl:Class rdf:about="&OntologyPL;J_2">
        <rdfs:label rdf:datatype="&xsd;string"
            >object-oriented programming languages</rdfs:label>
        <rdfs:subClassOf rdf:resource="&OntologyPL;J_1"/>
    </owl:Class>
    <!-- http://www.owl-ontologies.com/OntologyPL.owl#J_3 -->
    <owl:Class rdf:about="&OntologyPL;J_3">
        <rdfs:label rdf:datatype="&xsd;string">java</rdfs:label>
        <rdfs:subClassOf rdf:resource="&OntologyPL;J_2"/>
    </owl:Class>
</rdf:RDF>

<!-- Generated by the OWL API (version [Not Released]) http://owlapi.sourceforge.net -->
```

Приложение 2. Онтологии OntoJavaCSharp.owl и OntoPLFull.owl.

OntoJavaC#.owl

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
    xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:swrl="http://www.w3.org/2003/11/swrl#"
    xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
    xmlns="http://www.owl-ontologies.com/Ontology1265394254.owl#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://www.owl-ontologies.com/Ontology1265394254.owl">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="OJC_1">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >programing language</rdfs:label>
  </owl:Class>
  <owl:Class rdf:ID="OJC_4">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >object oriented programming language</rdfs:label>
    <rdfs:subClassOf rdf:resource="#OJC_1"/>
  </owl:Class>
```

```
  <owl:Class rdf:ID="OJC_2">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >java</rdfs:label>
    <rdfs:subClassOf rdf:resource="#OJC_4"/>
  </owl:Class>
  <owl:Class rdf:ID="OJC_3">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >C#</rdfs:label>
    <rdfs:subClassOf rdf:resource="#OJC_4"/>
  </owl:Class>
</rdf:RDF>
<!-- Created with Protege (with OWL Plugin 3.4.1, Build 536)  http://protege.stanford.edu →
OntoPLFull.owl
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
    xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
    xmlns="http://www.owl-ontologies.com/Ontology1265394087.owl#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:swrl="http://www.w3.org/2003/11/swrl#"
    xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://www.owl-ontologies.com/Ontology1265394087.owl">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="Class_8">
```

```
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >Scala</rdfs:label>
  <rdfs:subClassOf>
    <owl:Class rdf:ID="Class_5"/>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Class rdf:ID="Class_2"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Class_6">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >erlang</rdfs:label>
  <rdfs:subClassOf>
    <owl:Class rdf:about="#Class_5"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Class_7">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#Class_5"/>
  </rdfs:subClassOf>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >prolog</rdfs:label>
</owl:Class>
<owl:Class rdf:about="#Class_5">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >functional programming language</rdfs:label>
  <rdfs:subClassOf>
```

```
    <owl:Class rdf:ID="Class_1"/>
   </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Class_4">
   <rdfs:subClassOf>
    <owl:Class rdf:about="#Class_2"/>
   </rdfs:subClassOf>
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >C#</rdfs:label>
  </owl:Class>
  <owl:Class rdf:about="#Class_1">
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >programming language</rdfs:label>
  </owl:Class>
  <owl:Class rdf:about="#Class_2">
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >object-oriented programming language</rdfs:label>
   <rdfs:subClassOf rdf:resource="#Class_1"/>
  </owl:Class>
  <owl:Class rdf:ID="Class_3">
   <rdfs:subClassOf rdf:resource="#Class_2"/>
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >java</rdfs:label>
  </owl:Class>
</rdf:RDF>
<!-- Created with Protege (with OWL Plugin 3.4.1, Build 536)  http://protege.stanford.edu -->
```

Приложение 3. Онтологии OntoDrink.owl

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
    xmlns="http://www.owl-ontologies.com/Ontology1265394006.owl#"
    xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:swrl="http://www.w3.org/2003/11/swrl#"
    xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://www.owl-ontologies.com/Ontology1265394006.owl">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="OD_1">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >drinkable</rdfs:label>
  </owl:Class>
  <owl:Class rdf:ID="OD_3">
    <rdfs:subClassOf rdf:resource="#OD_1"/>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Cola</rdfs:label>
  </owl:Class>
  <owl:Class rdf:ID="OD_2">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >java</rdfs:label>
```

<rdfs:subClassOf rdf:resource="#OD_1"/>

</owl:Class>

</rdf:RDF>

<!-- Created with Protege (with OWL Plugin 3.4.1, Build 536) http://protege.stanford.edu -->