

**Министерство образования и науки Российской Федерации**  
**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**  
**“САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**  
**УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ,**  
**МЕХАНИКИ И ОПТИКИ”**

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**РАЗРАБОТКА ИНФОРМАЦИОННО-ПОИСКОВОЙ СИСТЕМЫ**  
**НАУЧНОГО И ОБРАЗОВАТЕЛЬНОГО КОНТЕНТА НА ОСНОВЕ**  
**ОТКРЫТЫХ ДАННЫХ**

Автор Герасин Олег Владимирович \_\_\_\_\_  
(Фамилия, Имя, Отчество) (Подпись)  
Направление подготовки (специальность) \_\_\_\_\_  
09.04.04 – Программная инженерия  
(код, наименование)  
Квалификация магистр \_\_\_\_\_  
(бакалавр, магистр)  
Руководитель Радченко И.А., доцент, к.т.н. \_\_\_\_\_  
(Фамилия, И., О., ученое звание, степень) (Подпись)

**К защите допустить**

Зав. кафедрой Муромцев Д. И., доцент, к. т. н. \_\_\_\_\_  
(Фамилия, И., О., ученое звание, степень) (Подпись)  
“ ” 20 \_\_\_\_ г.

Санкт-Петербург, 2018 г.

Студент Герасин О.В. Группа Р4217 Кафедра ИПМ Факультет ПИ и КТ  
(Фамилия, И., О.)

Направленность (профиль), специализация \_\_\_\_\_

09.04.04 – Разработка программно-информационных систем

Консультант (ы):

а) Навроцкий М.А., аспирант \_\_\_\_\_ (Подпись)  
(Фамилия, И., О., ученое звание, степень)

б) \_\_\_\_\_ (Подпись)  
(Фамилия, И., О., ученое звание, степень)

ВКР принята “ \_\_\_\_\_ ” \_\_\_\_\_ 20 \_\_\_\_\_ г.

Оригинальность ВКР \_\_\_\_\_ %

ВКР выполнена с оценкой \_\_\_\_\_

Дата защиты “ \_\_\_\_\_ ” \_\_\_\_\_ 20 \_\_\_\_\_ г.

Секретарь ГЭК \_\_\_\_\_ (Подпись)  
(Фамилия, И., О.)

Листов хранения \_\_\_\_\_

Демонстрационных материалов / чертежей хранения \_\_\_\_\_

## Оглавление

Введение.....	6
1 Анализ предметной области.....	9
1.1 Открытые данные.....	9
1.2 Связанные данные.....	11
1.3 Связанные открытые данные .....	13
1.4 Источники связанных открытых данных .....	14
1.4.1 Linked Open Aalto Data Service .....	15
1.4.2 Linked Data from The Open University .....	16
1.4.3 The EU Open Data Portal .....	17
1.4.4 Springer Nature SciGraph Data Explorer .....	19
1.5 Постановка задачи.....	20
2 Проектирование и разработка онтологической модели.....	22
2.1 Анализ возможности повторного использования существующих онтологий .....	22
2.1.1 Bibliographic Ontology Specification .....	23
2.1.2 Dublin Core Metadata Initiative.....	25
2.1.3 FOAF.....	28
2.2 Определение классов и свойств.....	32
2.3 Реализация онтологии в системе Protégé.....	32
3 Проектирование и разработка информационно-поисковой системы.....	35
3.1 Функциональные требования к разрабатываемому программному решению .....	35
3.2 Нефункциональные требования к разрабатываемому программному решению .....	36

3.3 Разработка проектных решений .....	37
3.3.1 Концептуальный уровень .....	38
3.3.2 Логический уровень .....	42
3.3.3 Физический уровень .....	43
3.4 Разработка программного интерфейса.....	45
4 Экспериментальное исследование информационно-поисковой системы.....	50
4.1 Тестирование системы .....	50
4.2 Пример решения задачи .....	51
4.3 Научная и практическая значимость работы .....	54
Заключение .....	56
Список используемых источников.....	58

## ВВЕДЕНИЕ

Развитие таких направлений, как машинное обучение, большие данные, невозможно без большого количества данных, доступных для анализа [1]. В связи с этим, актуальна проблема доступа к информации, доступной для машинной обработки. Ведь к настоящему времени накоплен существенный объем полезной информации, представленной в электронном виде, вот только возможности по ее использованию ограничены. Развитие такого направления, как открытые данные, призвано решить эту проблему.

Со временем появилась необходимость извлекать из данных дополнительную информацию, которая на первый взгляд неочевидна. Таким образом, появились связанные данные – метод публикации структурированных данных, связанных между собой.

Когда связанные данные публикуются в открытом доступе и соответствуют требованиям, предъявляемым к открытым данным, они называются связанными открытыми данными (Linked Open Data, LOD) [2]. Научное сообщество, бесспорно, играет ключевую роль в развитии этого направления [3]. Например, многие университеты интегрировались в международный образовательный процесс путем предоставления точек доступа к связанным открытым данным (в рамках европейского проекта Linked Universities) [4]. Для извлечения информации из них используются структурированные запросы SPARQL [5]. Но для обычного пользователя этот язык кажется сложным и, таким образом, является естественным барьером для использования связанных открытых данных.

Часть точек доступа проекта Linked Universities имеют общие предметные области (например, публикации), однако для того, чтобы получить результаты определенного поискового запроса, нужно выполнить запрос поочередно к каждой из этих точек доступа. Для этого необходим инструмент централизованного поиска по этим точкам, предоставляющий общую страницу результатов.

Таким образом, задача разработки информационно-поисковой системы, предоставляющей возможность централизованного поиска научных и образовательных данных, является актуальной.

В работе исследуются вопросы сбора информации с разных источников и создания собственных научно-образовательных ресурсов на основе связанных открытых данных.

Объектом являются научные и образовательные открытые данные – научные публикации.

Предметом исследования является процесс поиска и публикации связанных открытых научных и образовательных данных.

Целью данной работы является сбор информации по публикациям с помощью разрабатываемого единого инструмента централизованного поиска по источникам связанных открытых данных, в том числе, предоставляемых университетами, а также формирование собственной онтологической модели представления данных.

Практическая значимость работы заключается в наличии программного решения, которое позволит упростить процесс работы со связанными открытыми данными для пользователей, а также предоставит программный интерфейс, позволяющий другим системам обрабатывать результаты поисковых запросов. Данная работа также направлена на увеличение популярности и стимуляцию развития такого перспективного направления, как связанные открытые данные.

Для достижения поставленной цели требуется решение следующих задач:

1. Провести анализ предметной области (пространства связанных открытых данных), анализ источников связанных открытых данных, которые предоставляют информацию о публикациях, а также определить методы и модели, применимые для публикации научных и образовательных данных.

2. Спроектировать и реализовать онтологическую модель для хранения информации о публикациях.

3. Спроектировать и реализовать информационно-поисковую систему, обеспечивающую сбор данных о публикациях с разных источников и формирование собственной онтологической модели.

4. Провести экспериментальное исследование разработанного программного решения для сбора и хранения научных и образовательных данных.

## 1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

### 1.1 Открытые данные

Открытые данные (Open data) – это идея, что определённые данные должны быть свободно доступны для дальнейшей републикации и машиночитаемого использования без ограничений авторского права, патентов и других механизмов контроля [6]. Требования распространять их на тех же условиях, что и исходные, и указывать источник данных допустимы. Освободить данные от ограничений авторского права можно с помощью публичных лицензий. Если данные не доступны по лицензии, дающей повторным пользователям соответствующие права, или как общественное достояние, то они не являются открытыми данными, даже если выложены в машиночитаемом виде в Интернет.

Можно выделить следующие основные принципы открытых данных [7]:

- первичность;
- полнота;
- актуальность;
- лицензионная чистота;
- отсутствие дискриминации по доступу;
- пригодность к машинной обработке;
- отсутствие проприетарных форматов.

В последние годы для изучения потенциала рассматриваемой концепции [8] были разработаны различные порталы, предоставляющие доступ к открытым данным, такие как национальные открытые порталы данных, Европейская открытая инфраструктура данных [9], Junag [10], структуры статистических агентств (например, Eurostat[11]).

Наука основана на повторном использовании опубликованного научного знания. Для того, чтобы она могла эффективно функционировать, и чтобы общество могло воспользоваться всеми преимуществами научных усилий, крайне важно, чтобы научные данные были открыты.



Открытые научные данные предоставляют возможность совместного использования мировым сообществом. В связи с этим, в 2010 году были представлены Пантонские принципы [12], которым должны соответствовать научные данные для признания их открытыми:

Публиковать данные и коллекции данных необходимо с ясным и явным заявлением о пожеланиях и ожиданиях издателей в отношении повторного использования отдельных элементов данных, всей коллекции или её подмножества. Заявление должно быть точным и должно основываться на соответствующем и признанном юридическом заявлении в форме отказа или лицензии.

Многие широко признанные лицензии не предназначены и не подходят для данных или коллекций данных. Представлены различные лицензии, которые предназначены для публикации открытых данных [13]. Лицензии Creative Commons (кроме CCZero), GFDL, GPL, BSD не подходят для данных и их использование строго не рекомендуется.

Использование лицензий, ограничивающих коммерческое повторное использование или ограничение производства производных работ путем исключения использования для конкретных целей, настоятельно не рекомендуется. Эти лицензии не позволяют эффективно интегрировать наборы данных, а также предотвращают коммерческую деятельность, которая может быть использована для поддержки сохранения данных.

Кроме того, в науке настоятельно рекомендуется, чтобы данные, особенно в тех случаях, когда они публично финансировались, были явно размещены как общественное достояние. Это согласуется с государственным финансированием значительных научных исследований и общим характером совместного использования и повторного использования в рамках научного сообщества.

## 1.2 Связанные данные

Связанные данные (Linked data) – это метод публикации связанных между собой структурированных данных [14; 15]. Он основан на стандартных веб-технологиях, таких как HTTP, RDF и URI, но вместо того, чтобы использовать их для отображения веб страниц для пользователей, он расширяет их, чтобы распространять информацию, которая будет пригодна к машинной обработке. Это позволяет подключать и запрашивать данные из разных источников.

В 2006 году сэр Тимоти Джон Бернерс-Ли выделяет четыре базовых принципа проектирования применительно к связанным данным [16]:

- Необходимо применять унифицированные идентификаторы ресурсов (URI) для однозначной идентификации объектов (элементов данных).
- Использовать стандартные HTTP URL-адреса URI таким образом, чтобы можно было осуществлять информационный поиск.
- Предоставлять метаданные в соответствии с такими открытыми стандартами, как RDF.
- При публикации данных включать в описание ссылки на соответствующие URI других сущностей при наличии взаимосвязей для возможности обнаружения большего количества информационных объектов.

Связанные данные являются основой семантического веба (Semantic web) – они предоставляют широкомасштабную интеграцию и анализ данных в интернете. Такие возможности – это результат использования стандартного формата представления благодаря модели Resource Description Framework (RDF). RDF — это разработанная консорциумом W3C модель для описания ресурсов, в особенности — метаданных о ресурсах. В RDF данные организованы в триплетах, построенных на выражениях “субъект-предикат-объект”, как показано на рисунке 1.

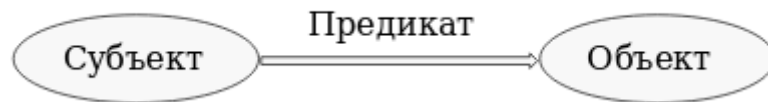


Рисунок 1 – RDF-триплет

Множество RDF-триплетов образует ориентированный граф, в котором вершинами являются субъекты и объекты, а ребра отображают отношения. Используя RDF можно логически выводить новые факты, а поиск в модели может быть организован с помощью языка запросов SPARQL. Пример запроса представлен на рисунке 2.

```

PREFIX bibo: <http://purl.org/ontology/bibo/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT
distinct ?URI ?title ?authors ?date ?abstract
FROM <http://data.open.ac.uk/context/oro>
WHERE {
    ?URI a bibo:Article ;
    bibo:authorList ?authors ;
    bibo:abstract ?abstract ;
    dcterms:title ?title ;
    dcterms:date ?date .
FILTER (regex( str(?title), "keyword", "i")).
}
GROUP BY ?URI ?title ?authors ?date ?abstract
  
```

Рисунок 2 – Пример SPARQL-запроса

Технология связанных данных удобна для интеграции данных, хранящихся в различных файловых системах, базах данных за счет используемой распределенной модели данных. Для связанных данных определен единый интерфейс, универсальный для всех приложений. Благодаря HTTP URL существует универсальная схема адресации для доступа и идентификации объектов. Простая и расширяемая модель данных RDF определяет такие важные характеристики технологии, как распределенность и масштабируемость.

В 2011 г. был запущен проект LinkedScience.org, в рамках которого было дано определение связанной науки [17]. Связанная наука (Linked Science) – это подход к объединению научных активов для обеспечения прозрачных,

воспроизводимых и трансдисциплинарных исследований. Это комбинация из четырех принципов:

1. Публикация научных данных, метаданных, результатов и информации о происхождении с использованием принципов связанных данных;
2. Использование окружений с открытым исходным кодом и веб-средой для выполнения, проверки и изучения исследований;
3. Облачные вычисления для эффективных и распределенных вычислений;
4. Creative Commons для правовой инфраструктуры.

### **1.3 Связанные открытые данные**

Определение связанных открытых данных (Linked Open Data, LOD) основывается на определениях открытых и связанных данных. Это связанные наборы данных, распространяемые под открытой лицензией и соответствующие требованиям, предъявляемым к открытым данным.

Организации, публикуя такую информацию и используя общепринятые пространства имён, присоединяются к международному движению Linked open data, сокращенно LOD. Множество наборов связанных открытых данных образуют пространство LOD, изображенное на рисунке 3.

Для классификации открытых данных сэр Тим Бернерс-Ли предложил рейтинговую систему:

1. Одна звезда – опубликованы в интернете в любом формате, но под открытой лицензией.
2. Две звезды – структурированные данные, доступные в машиночитаемом формате.
3. Три звезды – как две звезды, но формат данных должен быть непроприетарный.
4. Четыре звезды – как три звезды, но используются открытые стандарты, рекомендуемые консорциумом W3C (RDF, SPARQL) для идентификации.

5. Пять звезд – вдобавок к вышесказанному, данные должны быть связаны с другими наборами для обеспечения контекста.

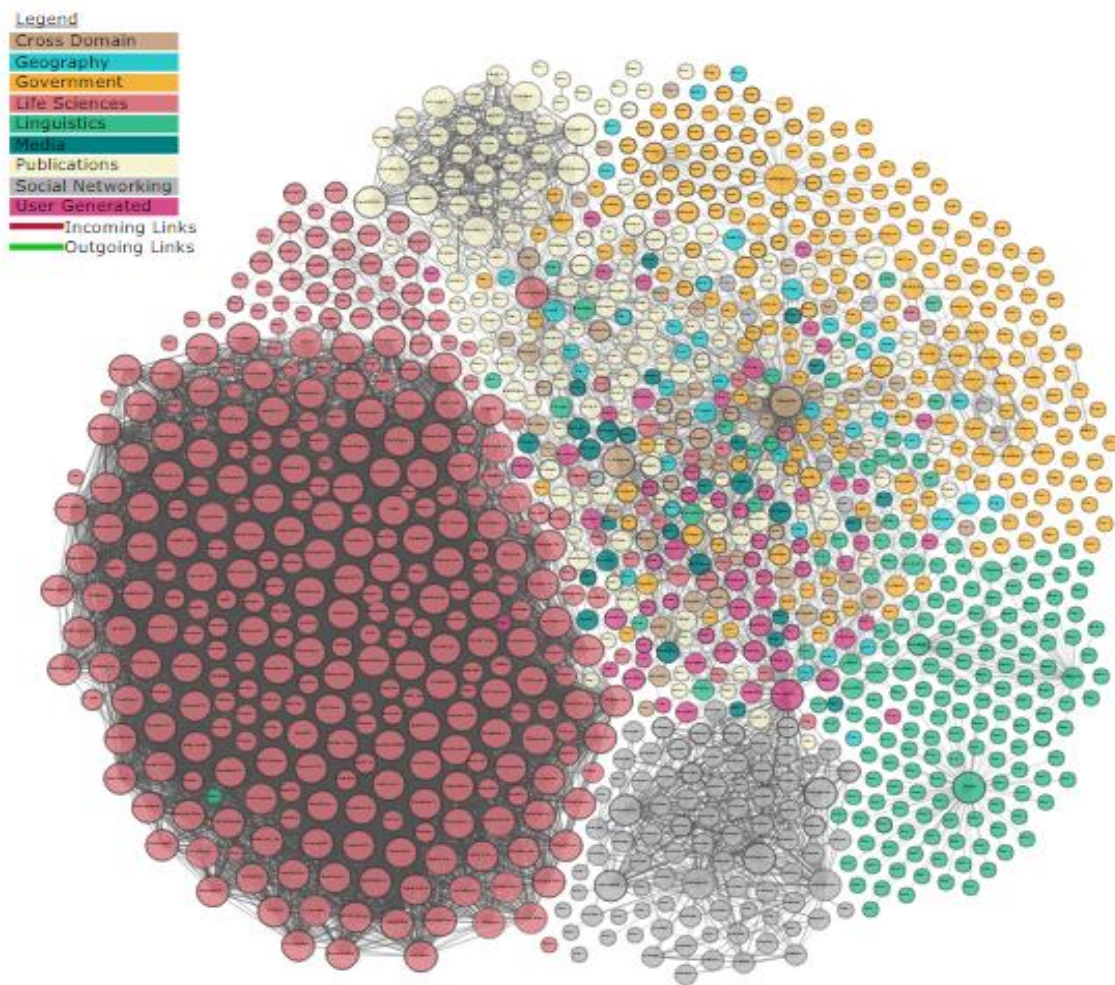


Рисунок 3 – Пространство LOD

Данные, интегрированные в пространство LOD, соответствуют пяти звездам по представленной шкале и являются наиболее предпочтительными и удобными для последующей обработки. Также они удовлетворяют вышеназванным Пантонским принципам и являются актуальным инструментом для использования в научных целях [18].

#### 1.4 Источники связанных открытых данных

Научное сообщество не осталось в стороне, и университеты, безусловно, играют ключевую роль в развитии этого направления. Например, в рамках

европейского проекта связанных университетов [LinkedUniversities.org](http://LinkedUniversities.org) университеты выкладывают свои наборы связанных открытых данных. Рассмотрим, какие источники связанных открытых данных предоставляют наборы публикаций.

#### **1.4.1 Linked Open Aalto Data Service**

Linked Open Aalto Data Service [19] предоставляет открытые связанные данные, опубликованные Университетом Аалто. Для доступа к ним организована точка доступа SPARQL, данные предоставляются как в машинном формате, так и в формате, доступном для браузера. Также возможна загрузка целых наборов данных.

Публикуемые наборы данных включают в себя информацию о:

- курсах;
- публикациях;
- исследовательских проектах;
- исследователях и сотрудниках;
- организационной структуре;
- новостях и событиях.

Данные могут использоваться для создания визуализаций и приложений, которые демонстрируют, как связанные открытые данные могут использоваться для повышения прозрачности университета, повышения ценности как для студентов, так и для персонала, а также для того, чтобы помочь лицам, принимающим решения, понять и отслеживать работу распределенных знаний на основе организации.

Этот сайт был создан в рамках проекта Linked Open Aalto в Semantic Computing Research Group, в сотрудничестве с [LinkedUniversities.org](http://LinkedUniversities.org). Университет повторно использует существующие словари Linked Data, чтобы поддерживать совместимость с данными других университетов по всему миру.

Данные публикаций доступны из двух баз данных: TKKjulkaisee и RESCAT. Данные были преобразованы в формат связанных данных, указанный в онтологии BIBO, с использованием также терминов Dublin Core и FOAF.

Примеры публикаций:

Diversity and Dissent in the Social Sciences: The Case of Organization Studies (из базы данных TKKjulkaise) – <http://data.aalto.fi/page/id/publication/rescat/rec8346>.

Fatigue Assessment of Laser Stake–Welded T-Joints (из базы данных RESCAT) – <http://data.aalto.fi/page/id/publication/tkkjulkaisee/rec65009>.

Доступные поля публикаций представлены в таблице 1.

Таблица 1 – Доступные поля публикаций в Linked Open Aalto Data Service

Название поля	Название отношения в RDF
Список авторов	bibo:authorList
Внесли вклад	?:contributor
Дата публикации	dc:date
Название	dc:title
Тип сущности	rdf:type

#### 1.4.2 Linked Data from The Open University

Открытый университет Великобритании публикует свои данные [20], имеющиеся в различных институциональных хранилищах Университета, связывает их и открывает для повторного использования. Данные могут запрашиваться через точку доступа SPARQL. В настоящее время представлены следующие наборы данных:

- публикации;
- квалификации;
- курсы и аудио/видео материалам, выпускаемым в Открытом университете;
- сотрудники, которые участвуют в разработке.

Все эти данные доступны через стандартные форматы (RDF и SPARQL) и в большинстве случаев доступны по открытой лицензии (Creative Commons Attribution 3.0 Unported License) [21].

Система Open Research Online содержит информацию о публикациях, предоставленную исследователями университетами. Это информация публикуется с использованием онтологии для библиографии (BIBO).

Пример публикации:

Semantic learning narratives – <http://data.open.ac.uk/page/oro/23658>. Доступные поля для публикаций представлены в таблице 2.

Таблица 2 – Доступные поля публикаций в Open University

Название поля	Название отношения в RDF
Тип сущности	rdf:type
Описание	bibo:abstract
Было представлено	bibo:presentedAt
В датасете	void:inDataset
Тип сущности	dcterms:creator
Название	dcterms:title
Статус	bibo:status
Авторы	bibo:authorList
Является частью	dcterms:isPartOf
Дата	dcterms:date
Метка	rdf:label

### 1.4.3 The EU Open Data Portal

Портал открытых данных Европейского союза (EU ODP) обеспечивает доступ к расширенному спектру данных из институтов Европейского союза (ЕС)



и других органов ЕС [22]. Разрешается использовать эти данные в коммерческих или некоммерческих целях.

Предоставляя легкий доступ к открытым данным, портал стремится помочь внедрить их в инновационное использование и разблокировать их экономический потенциал. Портал также предназначен для того, чтобы сделать институты ЕС и другие органы более открытыми и подотчетными.

Данные включают в себя:

- географические, геополитические и финансовые данные;
- статистика;
- результаты выборов;
- правовые акты;
- данные о преступности, здоровье, окружающей среде, транспорте;
- научные исследования.

Только институты, агентства и органы ЕС могут предоставлять данные для Открытого портала данных ЕС. Поставщики данных ЕС предоставляют portalу информацию, описывающую наборы данных, опубликованные на веб-сайте своего учреждения, агентства или органа ЕС.

Они могут предоставлять метаданные вручную через область поставщика данных или они могут отправлять данные автоматически в формате RDF.

Метаданные хранятся на портале. Он используется для перенаправления пользователей на веб-сайт поставщика данных, где они могут получить доступ к данным.

Также ведется работа над решением для извлечения данных с веб-сайтов автоматически.

Пример публикации:

Mind the lexical gap: EuroVoc : building block of the semantic web. – <http://publications.europa.eu/resource/cellar/ad2fb5f9-c5d4-4484-baab-0275698c36ec>.

Таблица 3 – Доступные поля публикаций в The EU Open Data Portal

Название поля	Название отношения в RDF
---------------	--------------------------

Тип сущности	rdf:type
Дата публикации	cdm:work_date_document
Идентификатор	cdm:work_id_document
Автор	cdm:work_created_by_agent
Название	cdm:expression_title

#### 1.4.4 Springer Nature SciGraph Data Explorer

Springer Nature SciGraph [23] – это проект, который объединяет источники данных от Springer Nature и ключевых партнеров из научного домена. Платформа предоставляет связанные открытые данные, объединяя информацию со всего исследовательского пространства, например, спонсоров, исследовательских проектов, конференций, филиалов и публикаций.

Платформа предоставляет высококачественные данные из надежных источников, которые обеспечивают богатое семантическое описание того, как связана информация, а также позволяют создавать инновационные визуализации научного домена.

Таким образом, Springer Nature SciGraph связывает всеобъемлющую информацию об исследовательском мире. Он представляет собой еще один шаг в интеграции данных и будет продолжать расти органично. Цель проекта – повысить открытость высококачественных данных, поскольку более широкие части наших наборов данных становятся доступными в рамках лицензирования CC-BY.

По прогнозам, данные в Springer Nature SciGraph будут содержать от 1,5 до 2 млрд RDF-триплетов. Постоянно добавляются дополнительные метаданные из журналов и статей, книг и глав, организаций, учреждений, исследовательских грантов, патентов, клинических испытаний, веществ, конференций, событий.

Пример публикации:

A semantic lexicon for medical language processing –  
<http://scigraph.springernature.com/things/articles/cd344429aa7ac2e5b9483ba53a8144fc#table>.

Таблица 4 – Доступные поля публикаций в Springer Nature SciGraph

Название поля	Название отношения в RDF
Тип сущности	rdf:type
Описание	sg:abstract
Цифровой идентификатор объекта	sg:doi
Ссылка на идентификатор	sg:doiLink
Спонсирован	sg:isFundedPublicationOf
Язык	sg:language
Лицензия	sg:license
Год публикации	sg:publicationYear
Название	sg:title
Метка	rdfs:label

### 1.5 Постановка задачи

Для поиска публикаций представленные выше источники данных зачастую предоставляют только SPARQL-редактор, и для того, чтобы найти определенную публикацию, нужно знать специфичный язык запросов SPARQL. Но для обычного пользователя этот язык кажется сложным и, таким образом, является естественным барьером для использования связанных открытых данных.

Рассмотренные выше источники имеют общие предметные области (например, публикации), однако для того, чтобы получить результаты определенного поискового запроса, нужно выполнить запрос поочередно к каждой из этих точек доступа. Для этого необходим инструмент

централизованного поиска по этим источникам, предоставляющий общую страницу результатов.

Представленные ресурсы также являются не связанными между собой, таким образом, идентификатор автора из одного ресурса не соответствует идентификатору автора из другого ресурса. Для решения этой проблемы необходимо построение новой онтологии, которая решала бы эту проблему.

## **2 ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ОНТОЛОГИЧЕСКОЙ МОДЕЛИ**

Для представления связанных открытых данных в машиночитаемом формате, необходимо определить все возможные виды объектов, их свойств и связей между ними. Для этого и для формализации определенной области знаний используются онтологии. Онтология служит для обеспечения общего словаря для обработки информации как машиной, так и человеком. Компонентами онтологии являются:

- понятий предметной области и их атрибуты;
- отношения между понятиями;
- аксиомы и правила вывода.

При построении онтологии нужно:

- определить области онтологии;
- по возможности повторно использовать существующие онтологии;
- перечислить важные термины в онтологии;
- определить классы;
- определить свойства классов.

В данной работе необходимо разработать систему, производящую поисковые запросы к точкам доступа связанных открытых данных, содержащим информацию о публикациях. Соответственно, разрабатываемая онтология будет использоваться для хранения самих публикаций и связанных с ними сущностей.

### **2.1 Анализ возможности повторного использования существующих онтологий**

Существует достаточно много онтологий, охватывающих тему публикаций. Для обеспечения совместимости с уже существующими онтологиями принято повторно использовать словари (или пространство имен), указанные в них. В онтологиях словари определяют отношения и понятия, используемые для

описания определенной области знаний. Проанализируем пространства имен, используемые в источниках, которые рассматриваются в данной работе.

- В онтологии от Университета Аалто используются словари <http://purl.org/ontology/bibo/> (bibo), [http://purl.org/dc/terms/\(dcterms\)](http://purl.org/dc/terms/(dcterms)), [http://xmlns.com/foaf/spec/ \(foaf\)](http://xmlns.com/foaf/spec/(foaf)).

- Открытый университет Великобритании пользуется теми же словарями, что и Университет Аалто, так как они оба участвуют в проекте связанных университетов.

- Портал открытых данных европейского союза использует в онтологии свои имена из <http://publications.europa.eu/ontology/cdm/> (cdm).

- Springer Nature SciGraph также используют свои имена из [http://scigraph.springernature.com/ontologies/core/ \(sg\)](http://scigraph.springernature.com/ontologies/core/(sg)).

Остановимся подробнее на словарях bibo, dcterms и foaf.

### **2.1.1 Bibliographic Ontology Specification**

Bibliographic Ontology Specification (Bibo) – это библиографическая онтология, которая описывает библиографические данные о семантическом вебе в RDF [24]. Она может использоваться как онтология классификации документов, онтология цитирования или просто как способ описания любого вида документа в RDF. Основана на многих существующих форматах метаданных и может использоваться в качестве общей основы для преобразования других источников библиографических данных.

Данное пространство имен описывает библиографическую онтологию и термины (классы и свойства RDF), которые ее составляют, поэтому приложения, работающие с семантическим вебом, могут использовать эти термины в различных форматах и приложениях, совместимых с RDF. Данная онтология проста, прагматична и предназначена для широкомасштабного использования. Представлена на английском языке, разработчики – Bruce D’Arcus, Frédéric Giasson.

Определенные в данной онтологии свойства и типы предоставляют некоторые базовые концепции для использования в описаниях библиографической онтологии. Другие словари (например, элементы метаданных Dublin Core для простого библиографического описания, FOAF и т.д.) также могут быть смешаны с терминами Bibliographic Ontology. Библиографическая онтология предусматривает расширение, а также впоследствии к ней могут быть добавлены модули.

Библиографическая онтология использует фреймворк описания ресурсов (RDF), потому что предметная область, которая описывается в ней – цитаты и библиографические ссылки, имеет так много конкурирующих требований, что какой-то отдельный формат не будет их содержать полностью или приведет к попытке описать эти требования в ряде несовместимых форматов. Используя RDF, Библиографическая Онтология имеет мощный механизм расширяемости, позволяя смешивать описания на основе библиографии и онтологии с утверждениями, сделанными в любом другом словаре RDF.

Библиографическая онтология не может включать в себя все, что связано с цитатами и библиографическими ссылками. Вместо того чтобы охватывать все темы в самой библиографической онтологии, разработчики описывают основные темы и встраивают в более крупную структуру RDF, которая позволяет использовать работу в другом месте для более конкретных словарей описания.

На рисунке 4 представлено описание стандартной статьи журнала с использованием словаря Bibo:

```
<info:doi/10.1134/S0003683806040089> a bibo:Article ;
  dc:title "Effect of argillaceous minerals on the growth of phosphate-mobilizing bacteria Bacillus subtilis"@en ;
  dc:date "2006-01-01" ;
  dc:isPartOf <urn:issn:23346587> ;
  bibo:volume "42" ;
  bibo:issue "4" ;
  bibo:pageStart "388" ;
  bibo:pageEnd "391" ;
  dc:creator <http://examples.net/contributors/2> ;
  dc:creator <http://examples.net/contributors/1> ;
  bibo:authorList ( <http://examples.net/contributors/2> <http://examples.net/contributors/1> ) .
```

Рисунок 4 – Описание статьи журнала

Класс статьи имеет тип `bibo:Article`, описание полей (свойств) представлено в таблице 5. Поле `bibo:authorList` является упорядоченным списком авторов и определено с использованием `rdf:List`.

Таблица 5 – Описание полей статьи журнала

Название поля	Название отношения в RDF
Название	<code>dc:title</code>
Дата	<code>dc:date</code>
Является частью	<code>dc:isPartOf</code>
Ссылка на идентификатор	<code>bibo:volume</code>
Спонсирован	<code>bibo:issue</code>
Язык	<code>bibo:pageStart</code>
Лицензия	<code>bibo:pageEnd</code>
Год публикации	<code>dc:creator</code>
Название	<code>bibo:authorList</code>

### 2.1.2 Dublin Core Metadata Initiative

Инициатива метаданных Дублинского ядра (Dublin Core Metadata Initiative, DCMI) – это открытая организация, поддерживающая инновации в разработке метаданных [25].

В рамках своей миссии Инициатива метаданных Дублинского ядра разрабатывает и поддерживает спецификации для описания ресурсов. Спецификациям, разработанные и рассмотренные в контексте процесса официального утверждения DCMI, присваивается один из статусов (в порядке возрастания зрелости и стабильности) «Рабочий проект DCMI», «Предлагаемая рекомендация DCMI» или «Рекомендация DCMI».



Хотя Дублинское ядро изначально было разработано с целью описания объектов, подобных документам (поскольку традиционные текстовые ресурсы достаточно хорошо обрабатывались машиной), метаданные DC могут быть применены и к другим ресурсам. Их пригодность для использования с конкретными ресурсами, не связанными с документами, будет зависеть в некоторой степени от того, насколько тесно их метаданные напоминают типичные метаданные документа, а также то, для чего они предназначены.

Dublin Core имеет следующие особенности:

- Простота создания и обслуживания;
- Понятая семантика;
- Международный охват;
- Расширяемость.

Рекомендация DCMI Metadata Terms<sup>1</sup> (Dcterms) представляет собой единый источник актуальной информации о терминологии метаданных DCMI, включая классический набор элементов метаданных Dublin Core, словарь типов DCMI и классы ресурсов. По сути, это словарь (пространство имен) основных понятий английского языка, унифицирующий метаданные для описания широкого диапазона ресурсов. С 2005 года словарь представлен и по модели RDF и является популярной основой для описания ресурсов в Семантической паутине. Семантика этого словаря была создана международной междисциплинарной группой профессионалов библиотечного дела, информатики, кодирования текстов, музейного сообщества и других смежных групп.

На рисунке 5 представлено простой пример, демонстрирующий описание аудиозаписи руководства по выращиванию кустов роз. Потенциально термины из словаря Дублинского ядра могут быть смешаны с другими словарями метаданных при необходимости.

---

<sup>1</sup> DCMI Metadata Terms. URL: <http://dublincore.org/documents/dcmi-terms/> (дата обращения: 25.04.2018)

```

<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/">

  <rdf:Description rdf:about="http://media.example.com/audio/guide.ra">

    <dc:creator>Rose Bush</dc:creator>
    <dc:title>A Guide to Growing Roses</dc:title>
    <dc:description>Describes process for planting and nurturing different kinds of rose bushes.</dc:description>
    <dc:date>2001-01-20</dc:date>

  </rdf:Description>
</rdf:RDF>

```

Рисунок 5 –Пример описание с использованием Dcterms

В таблице 6 приведены элементы простого набора метаданных Дублинского ядра.

Таблица 6 – Описание полей статьи журнала

Название поля	Название отношения в RDF
Название	dc:title
Создатель	dc:creator
Тема	dc:subject
Описание	dc:description
Издатель	dc:publisher
Внес вклад	dc:contributor
Дата	dc:date
Тип	dc:type
Формат документа	dc:format
Идентификатор	dc:identifier
Источник	dc:source
Язык	dc:language
Отношение	dc:relation
Покрытие	dc:coverage
Авторские права	dc:rights

### 2.1.3 FOAF

FOAF – это проект, посвященный связыванию информации о людях с использованием Интернета. Представлена на английском языке, разработчики: Dan Brickley, Libby Miller. FOAF объединяет три вида сети [26]:

- социальные сети человеческого сотрудничества;
- репрезентативные сети, описывающие упрощенное представление в фактических терминах;
- информационные сети, которые используют привязку к Интернету, для совместного использования независимо опубликованных описаний этого мира.

FOAF не конкурирует с социально ориентированными веб-сайтами; скорее, он обеспечивает подход, при котором разные сайты могут сообщать разные части более крупной истории, и с помощью которых пользователи могут сохранять некоторый контроль над своей информацией в непатентованном формате.

FOAF постепенно развивается с момента его создания в середине 2000 года. В настоящее время существует стабильное ядро классов и свойств, которые не будут изменены, кроме небольших корректировок их документации для отслеживания обратной связи. Новые термины могут быть добавлены в любое время (как в словаре на естественном языке), и, следовательно, эта спецификация является развивающейся работой. URI пространства имен FOAF RDF, напротив, является фиксированным, и его идентификатор, как ожидается, не изменится. Кроме того, предпринимаются усилия для обеспечения долгосрочного сохранения пространства имен FOAF, его доменного имени `xmlns.com` и соответствующей документации.

В описаниях FOAF существуют только различные виды и ссылки, которые называются свойствами. Типы вещей в FOAF называются классами, поэтому FOAF определяется как словарь терминов, каждый из которых является либо классом, либо свойством. Другие проекты наряду с FOAF предоставляют другие

наборы классов и свойств, многие из которых связаны с теми, которые определены в FOAF.

Описания FOAF публикуются как связанные документы в Интернете (например, с использованием синтаксиса RDF / XML или RDFa). Результатом проекта FOAF является сеть документов, описывающих сеть людей (и прочее). Каждый документ FOAF сам по себе является кодировкой описательной сетевой структуры. Хотя эти документы не всегда согласны, они обладают отличительной характеристикой – они могут быть легко объединены, что позволяет сочетать частичные и децентрализованные описания интересными способами.

FOAF объединяет множество терминов; некоторые описывают людей, некоторые группы, некоторые документы. Различные виды приложений могут использовать или игнорировать различные части FOAF. Ниже показан один из способов просмотра терминов FOAF: игнорирование архаичных и исторических деталей и разделение остальных на термины, которые имеют смысл только в Интернете, и те, которые имеют универсальную применимость при связывании людей и информации.

Категории основных терминов FOAF:

Core – эти классы и свойства образуют ядро FOAF. Они описывают характеристики людей и социальных групп, которые не зависят от времени и технологий; как таковые они могут быть использованы для описания базовой информации о людях в современном, историческом, культурном наследии и контексте цифровой библиотеки. В дополнение к различным характеристикам людей FOAF определяет классы для Project, Organization и Group как другие виды агентов.

Social Web – в дополнение к основным терминам FOAF существует ряд терминов для описания учетных записей Интернета, адресных книг и других веб-мероприятий.

Утилиты связанных данных – FOAF начинался как проект «RDFWeb» и создал широко распространенную модель для публикации простых фактических данных через связанные документы RDF. FOAF по-прежнему играет важную роль

в растущем сообществе «Связанные данные», акцентируя внимание на информации, которую затруднительно представить как простые фактические данные. FOAF – это попытка использовать Интернет для интеграции фактической информации с информацией в документах, ориентированных на человека (например, видео, книги, электронные таблицы, 3d модели), а также информации, которая все еще находится в головах людей. Это объясняет, почему FOAF включает несколько «демонстрационных» терминов, которые служили в основном образовательным целям (например, `geekcode`), наряду с несколькими техническими терминами (например, `focus`, `LabelProperty`), которые поддерживают более широкие усилия для связывания информации. Основанный на словарях дизайн FOAF допускает определенный прагматизм: разработчики просто записывают набор терминов, которые полезны для веб-сообщества, при этом акцентируя внимание на центральной идее FOAF, которая связана с увязкой сетей информации с сетями людей.

Основные классы и свойства FOAF представлены на рисунке 6.

FOAF Core	Social Web
<ul style="list-style-type: none"> <li>• <ul style="list-style-type: none"> <li>◦ <a href="#">Agent</a></li> <li>◦ <a href="#">Person</a></li> <li>◦ <a href="#">name</a></li> <li>◦ <a href="#">title</a></li> <li>◦ <a href="#">img</a></li> <li>◦ <a href="#">depiction</a> (<a href="#">depicts</a>)</li> <li>◦ <a href="#">familyName</a></li> <li>◦ <a href="#">givenName</a></li> <li>◦ <a href="#">knows</a></li> <li>◦ <a href="#">based_near</a></li> <li>◦ <a href="#">age</a></li> <li>◦ <a href="#">made</a> (<a href="#">maker</a>)</li> <li>◦ <a href="#">primaryTopic</a> (<a href="#">primaryTopicOf</a>)</li> </ul> </li> <li>• <ul style="list-style-type: none"> <li>◦ <a href="#">Project</a></li> <li>◦ <a href="#">Organization</a></li> <li>◦ <a href="#">Group</a></li> <li>◦ <a href="#">member</a></li> </ul> </li> <li>• <ul style="list-style-type: none"> <li>◦ <a href="#">Document</a></li> <li>◦ <a href="#">Image</a></li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>• <a href="#">nick</a></li> <li>• <a href="#">mbox</a></li> <li>• <a href="#">homepage</a></li> <li>• <a href="#">weblog</a></li> <li>• <a href="#">openid</a></li> <li>• <a href="#">jabberID</a></li> <li>• <a href="#">mbox_sha1sum</a></li> <li>• <a href="#">interest</a></li> <li>• <a href="#">topic_interest</a></li> <li>• <a href="#">topic</a> (<a href="#">page</a>)</li> <li>• <a href="#">workplaceHomepage</a></li> <li>• <a href="#">workInfoHomepage</a></li> <li>• <a href="#">schoolHomepage</a></li> <li>• <a href="#">publications</a></li> <li>• <a href="#">currentProject</a></li> <li>• <a href="#">pastProject</a></li> <li>• <a href="#">account</a></li> <li>• <a href="#">OnlineAccount</a></li> <li>• <a href="#">accountName</a></li> <li>• <a href="#">accountServiceHomepage</a></li> <li>• <a href="#">PersonalProfileDocument</a></li> <li>• <a href="#">tipjar</a></li> <li>• <a href="#">sha1</a></li> <li>• <a href="#">thumbnail</a></li> <li>• <a href="#">logo</a></li> </ul>

Рисунок 6 – Основные классы и свойства FOAF

На рисунке 7 показан простой документ, описывающий человека.

```
<foaf:Person rdf:about="#danbri" xmlns:foaf="http://xmlns.com/foaf/0.1/">
  <foaf:name>Dan Brickley</foaf:name>
  <foaf:homepage rdf:resource="http://danbri.org/" />
  <foaf:openid rdf:resource="http://danbri.org/" />
  <foaf:img rdf:resource="/images/me.jpg" />
</foaf:Person>
```

Рисунок 7 – Документ, описывающий человека

В этом кратком примере представлены основы FOAF. Он гласит: «Существует Человек (foaf:Person) с именем (foaf:name) «Dan Brickley», этот человек имеет домашнюю страницу (foaf:homepage) и foaf:openid «http://danbri.org/», а также изображение (foaf:img) с относительным URI «/images/me.jpg».

По популярности использования в онтологиях публикаций рассмотренные словари значительно превосходят другие. Поэтому для совместимости с другими

онтологиями, в разрабатываемой онтологии будут использоваться именно эти словари.

## 2.2 Определение классов и свойств

Предметная область будет строиться по двум классам:

- класс “Публикация” (Article);
- класс “Человек” (Person).

В таблице 5 дано описание свойств классов.

Таблица 7 – Список свойств в онтологии

Название	Принадлежит классу	Описание
bibo:uri	Article, Author	Идентификатор
dcterms:title	Article	Название
bibo:authorList	Article	Список авторов
dcterms:date	Article	Дата
bibo:abstract	Article	Описание
foaf:name	Author	Имя

## 2.3 Реализация онтологии в системе Protégé

В данной работе онтологическая модель представления знаний реализована в программном средстве Protégé [27]. К стандартным свойствам добавлены описанные выше, как показано на рисунке 8.

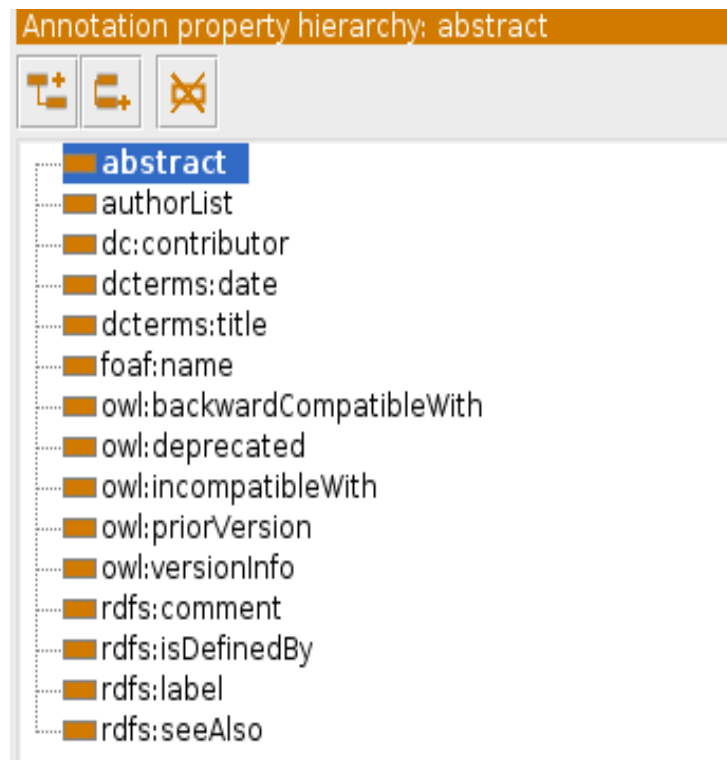


Рисунок 8 – Список свойств в онтологии

Диаграмма классов онтологии изображена на рисунке 9.

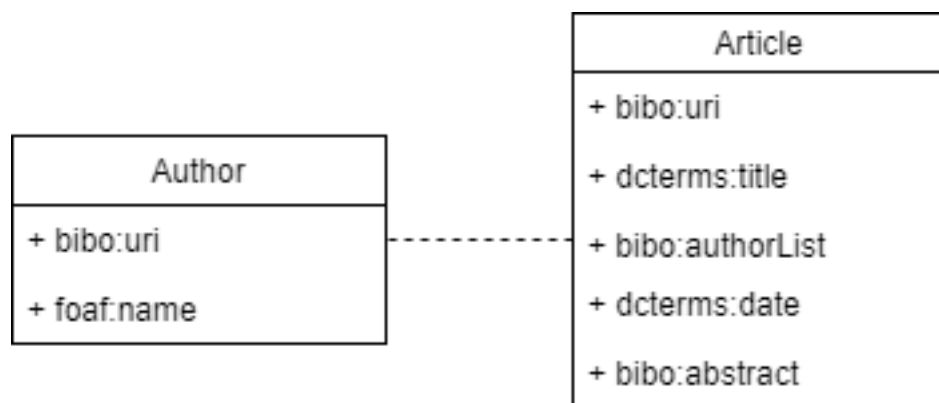


Рисунок 9 – Диаграмма классов онтологии

Были созданы необходимые классы, добавлен тестовый экземпляр класса Article, изображенный на рисунке 10.



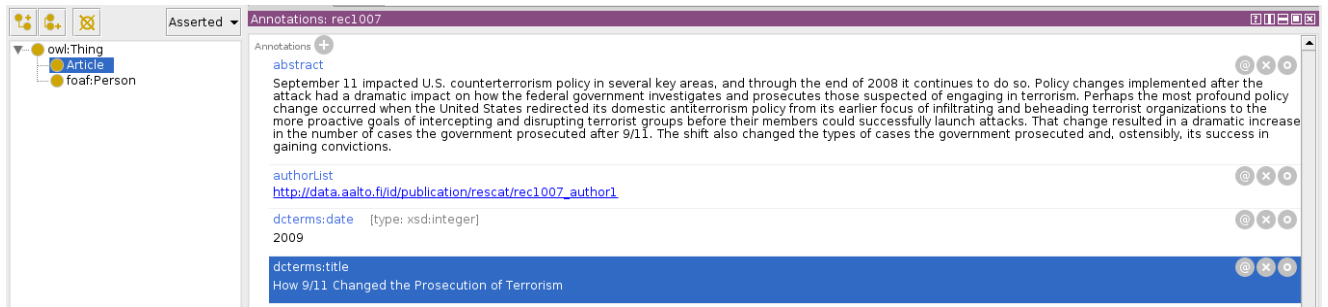


Рисунок 10 – Экземпляр класса Article в программном средстве Protege

### 3 ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ИНФОРМАЦИОННО-ПОИСКОВОЙ СИСТЕМЫ

#### 3.1 Функциональные требования к разрабатываемому программному решению

В рамках данной работы требуется разработать систему информационного поиска, которая должна осуществлять поиск среди связанных открытых данных и позволить пользователю единым образом взаимодействовать с публикациями из разных источников.

Основные требования к системе представлены на рисунке 11 в виде диаграммы вариантов использования.

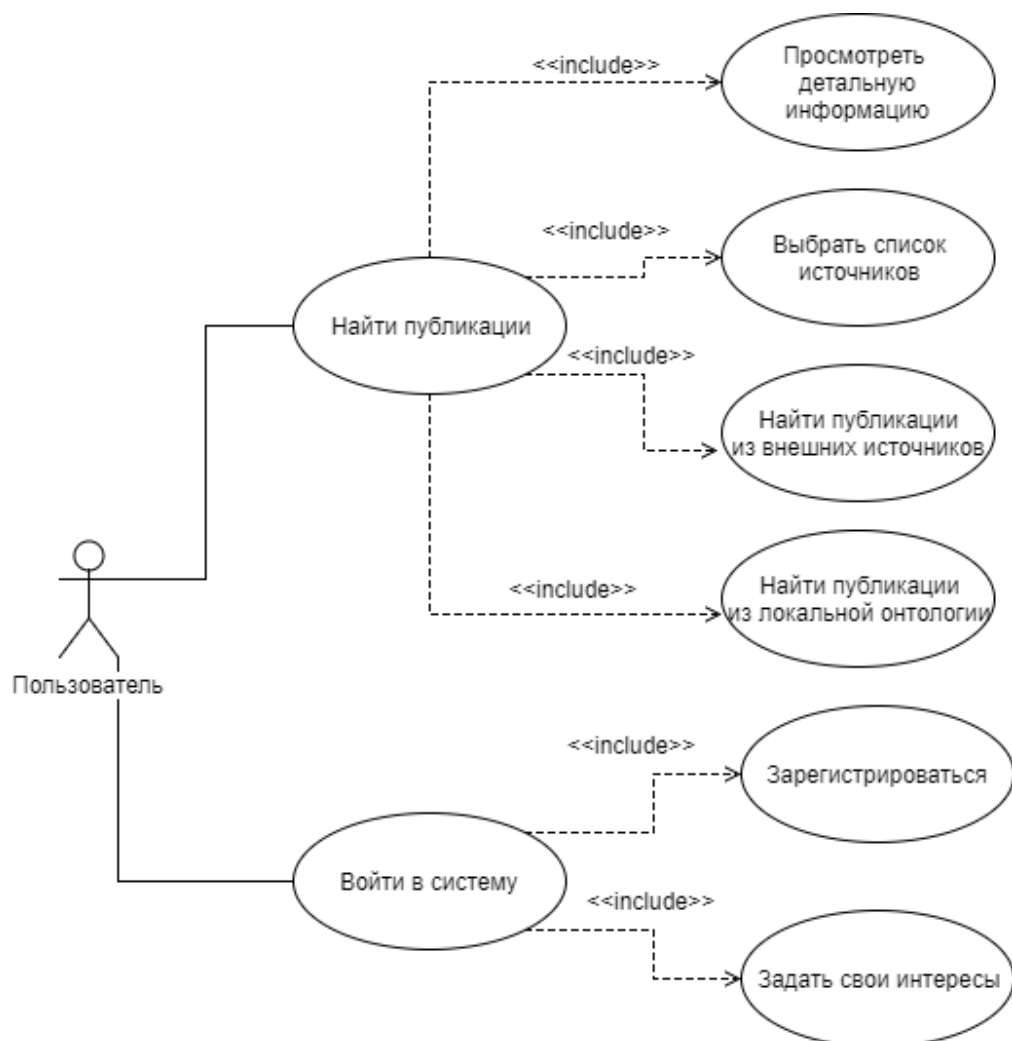


Рисунок 11 – Диаграмма вариантов использования разрабатываемой системы в нотации UML

Основные функции разрабатываемого программного решения:

1. Регистрация и авторизация пользователя;
2. Поиск среди источников связанных открытых данных;
3. Возможность выбора источников;
4. Кеширование;
5. Учет интересов пользователя;
6. Формирование онтологии на основе результатов запросов.

Регистрация и авторизация пользователя необходима для идентификации пользователя в системе и сохранения интересов для конкретного пользователя.

Возможность выбора источников данных и предоставление результатов прошлого запроса по тем же ключевым словам для моментального ответа пользователю (кеширование) позволяет сократить время ответа системы и учитывать индивидуальные требования пользователя.

Система учитывает интересы пользователя, выдавая наиболее интересные результаты выше остальных, что положительно сказывается на общем впечатлении от использования системы.

И, наконец, формирование онтологии на основе результатов запросов позволяет выводить новые факты, которые могут быть не очевидны на первый взгляд, а также объединяет данные из разных онтологий с разным описанием в одну, использующую стандартизированный словарь.

### **3.2 Нефункциональные требования к разрабатываемому программному решению**

Помимо функциональных требований выдвинуты следующие нефункциональные требования:

- система имеет клиент-серверную архитектуру;
- в качестве системы управления базой данных используется MongoDB;
- система реализуется на языке программирования Java 8, с использованием веб-фреймворка JSF 2.0;

- клиент реализуется в виде веб-приложения с использованием UI-фреймворка Primefaces;
- для доступа к онтологии используется программное средство Protege;
- файл, содержащий онтологическую модель предметной области, имеет формат owl 2.0;
- для запросов к источникам связанных открытых данных используется язык запросов SPARQL и библиотека Apache Jena;
- для использования результатов работы системы в других программных средствах предоставляется REST API.

### **3.3 Разработка проектных решений**

Для решения поставленной задачи разработана информационно-поисковая система научного и образовательного контента на основе связанных открытых данных. Система реализована на Java SE 8, Java EE 7 с использованием веб-фреймворка JSF версии 2.2.

Был выбран язык программирования Java, так как он является объектно-ориентированным, строго-типизированным и кроссплатформенным, за счет чего на нем удобно создавать приложения любой сложности. Это очень популярный язык программирования, и поэтому для него можно найти подключаемые библиотеки практически для любой цели. Разработчики языка добились элегантного решения для таких сложных задач, как многопоточность, сетевое взаимодействие, создание распределённых приложений, работа с потоками данных, предотвращение утечки памяти, перехватывание и обработка ошибок и исключений, что облегчает создание крупных и масштабируемых приложений.

На данном этапе работы были выделены проектные решения на трех уровнях разработки системы: концептуальном, логическом и физическом.

### 3.3.1 Концептуальный уровень

На концептуальном уровне была разработана архитектура системы и определен состав подсистем. Их описание приведено в таблице 5.

Таблица 8 – Описание подсистем

Название	Функции	Вход	Выход
Клиент	получение данных от пользователя, ввод/вывод контента	поисковой запрос, данные о интересах пользователя	публикации
Модуль вывода данных	формирование HTML-страниц для вывода на клиенте	научный и образовательный контент	сгенерированные HTML-страницы
Модуль получения публикаций	получение публикаций по поисковому запросу пользователя	поисковой запрос	найденные публикации
Модуль обработки публикаций	фильтрация публикаций по интересам пользователя	публикации	сгруппированные публикации
Модуль работы с онтологией	добавление в онтологию результатов запроса, взятие кэшированных результатов	научный и образовательный контент / поисковой запрос	научный и образовательный контент в owl-файле / результаты прошлого поискового запроса
Модуль работы с базой данных	хранение данных о пользователях	поисковой запрос	запись о пользователе

На рисунке 12 представлена архитектура разработанной системы.

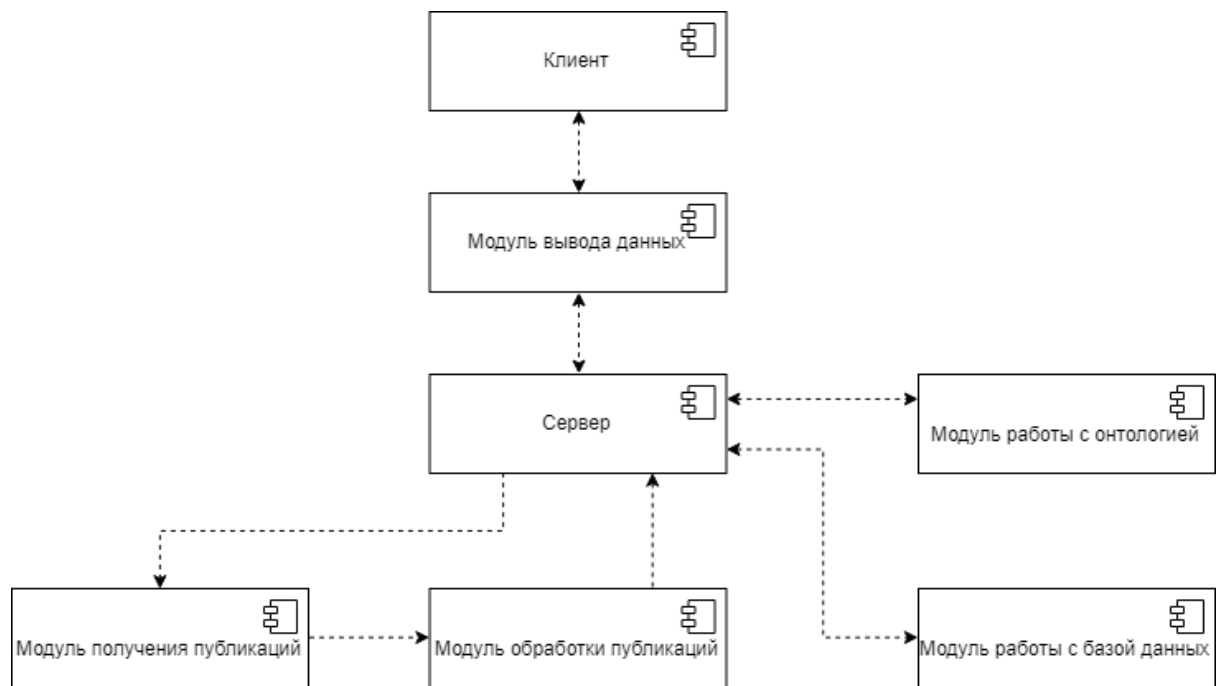


Рисунок 12 – Архитектура системы

Разработанная система состоит из следующих подсистем.

Клиент расположен на самом верхнем уровне, содержит сгенерированные HTML-страницы, отображаемые в браузере пользователя. Для реализации использовались следующие современные технологии:

- Java EE 7;
- Java Server Faces (JSF).

Java EE (Java Platform Enterprise Edition, Jakarta EE) представляет собой набор спецификаций и соответствующей документации для языка Java, описывающей архитектуру серверной платформы для задач предприятий. Основная целью спецификаций является обеспечение масштабируемости приложений и целостности данных во время работы системы. Java EE в основном ориентирована на использование ее через веб.

Веб-фреймворк Java Server Faces (JSF) служит для облегчения разработки компонентно-ориентированных пользовательских интерфейсов для Java EE приложений. В отличие от прочих MVC-фреймворков, которые управляются запросами, подход JSF основывается на использовании компонентов. Состояние компонентов пользовательского интерфейса сохраняется, когда пользователь

запрашивает новую страницу и затем восстанавливается, если запрос повторяется. Для отображения данных обычно используется JSP<sup>2</sup>, Facelets<sup>3</sup>.

Технология JavaServer Faces усиливает существующие стандартные концепции пользовательского интерфейса (UI) и концепции Web-уровня без привязки разработчика к конкретному языку разметки, протоколу или клиентскому устройству. Классы компонентов пользовательского интерфейса, поставляемые вместе с технологией JavaServer Faces, открывают возможность отображения JSF-компонент на различных клиентских устройствах, так как содержат функциональность компонент, а не специфичное для клиента отображение. Разработчики совмещают функциональность компонент интерфейса пользователя со специальными рендерерами, таким образом конструируя специальные теги для клиентского устройства. Технология JSF предоставляет специальную библиотеку JSP-тегов для рендеринга на HTML-клиенте и специфичный рендерер, позволяя разработчикам приложений на Java EE платформе использовать технологию JSF в своих приложениях.

Модуль вывода данных – это контроллер в терминологии модели MVC (Model-View-Controller), координатор трафика, который обрабатывает команды и входящие данные от пользователя, направляет их в нужные части и выбирает представление для отображения. В JSF этот контроллер представлен FacesServlet.

Модуль получения публикаций – класс для получения публикаций по введенному поисковому запросу. Для работы с точками доступа SPARQL используется библиотека Apache Jena.

Jena – это Java-платформа для создания семантических веб-приложений. Она предоставляет обширные библиотеки Java, помогающие разработчикам разрабатывать код, который обрабатывает RDF, RDFS, RDFa, OWL и SPARQL в соответствии с опубликованными рекомендациями W3C. Jena включает механизм вывода на основе правил для выполнения рассуждений на основе онтологий OWL

---

<sup>2</sup> JavaServer Pages Technology. URL: <http://www.oracle.com/technetwork/java/javaee/jsp/index.html> (дата обращения 14.04.2018).

<sup>3</sup> Introduction to Facelets. URL: <https://docs.oracle.com/javaee/7/tutorial/jsf-facelets.htm> (дата обращения 20.04.2018).

и RDFS, а также множество стратегий хранения для хранения RDF в памяти или на диске.

Jena была первоначально разработана исследователями в HP Labs, основана в Бристоле, Великобритания, в 2000 году. Jena всегда была проектом с открытым исходным кодом и сейчас широко используется в самых разных семантических веб-приложениях. В 2009 году HP решила переориентировать деятельность в области развития от непосредственной поддержки развития Jena и осталась в стороне от целей проекта. В ноябре 2010 года проект Jena был принят Apache Software Foundation.

Модуль обработки публикаций – класс для извлечения интересных публикаций из общего списка.

Модуль работы с онтологией – класс, предназначенный для занесения результатов поискового запроса в локальную онтологию.

Модуль работы с базами данных – класс, заносащий и извлекающий информацию из базы данных. В качестве системы управления базами данных используется MongoDB. MongoDB – это база данных документов, ориентированная на масштабируемость и гибкость.

MongoDB хранит данные в гибких JSON-подобных документах, что означает, что поля могут меняться от документа к документу и структуре данных со временем. Модель документа сопоставляется с объектами кода приложения, что упрощает работу с данными. Специальные запросы, индексирование и агрегирование в реальном времени предоставляют мощные способы доступа и анализа данных.

MongoDB – это распределенная база данных по своей сути, поэтому высокая доступность, масштабирование по горизонтали и географическое распределение встроены и просты в использовании. Она имеет бесплатный и открытый исходный код, опубликованный в соответствии с GNU Affero General Public License.

Модель документа MongoDB проста для изучения и использования разработчиками, при этом она обеспечивает все возможности, необходимые для



удовлетворения самых сложных требований в любом масштабе. Предоставляются драйверы для более 10 языков, а сообщество построило еще десятки.

Для связывания базы данных с концепциями объектно-ориентированных языков программирования существуют ORM (Object-Relational Mapping, объектно-реляционное отображение, преобразование) технологии. В концепциях СУБД MongoDB технологии, выполняющие такую же роль называются ODM (Object document mapping, преобразование документа в объект). В Java существует ODM Morphia<sup>4</sup>, которая и использовалась в данной работе. К преимуществам данной технологии можно отнести:

- проста в использовании;
- поддерживает как аннотированные POJO объекты, так и DAO подход;
- вся конфигурация задается аннотациями, XML файлы не используются;
- работает с Google Guice, Spring и другими DI фреймворками;
- имеется подробная документация, открытый исходный код и большое сообщество разработчиков.

### 3.3.2 Логический уровень

Были разработаны основные алгоритмы работы пользователя с системой. На рисунке 13 представлен основной алгоритм.

Система принимает от пользователя поисковой запрос, затем формирует SPARQL-запросы к точкам доступа связанных открытых данных. Сформированные запросы посылаются поочередно к источникам LOD, посредством чего собирается информация о публикациях. Система получает интересы пользователя из его профиля, с помощью них группирует публикации на две группы:

- В первую группу входят публикации, которые в своем описании содержат ключевые слова, которые пользователь ввел как интересы.
- Во второй группе находятся все остальные публикации.

---

<sup>4</sup> Morphia. URL: <https://mongodb.github.io/morphia/> (дата обращения 07.05.2018).



Рисунок 13 – Алгоритм поиска публикаций

### 3.3.3 Физический уровень

На физическом уровне была разработана диаграмма классов и механизмы взаимодействия между модулями системы.

Диаграмма классов системы приведена на рисунке 14. Для ясности на диаграмму вынесены лишь основные классы.

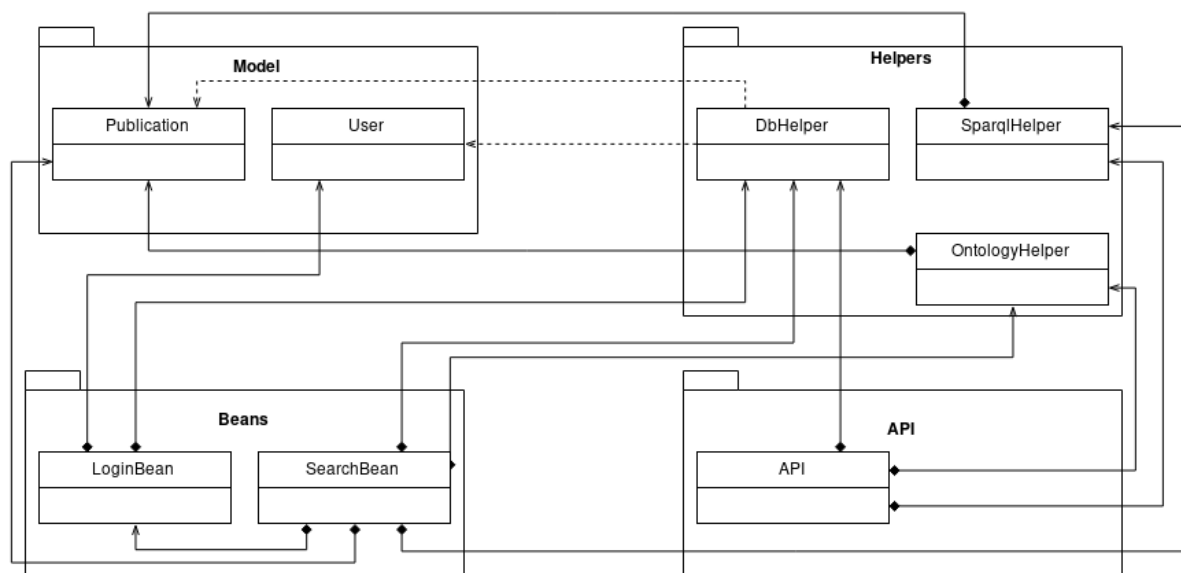


Рисунок 14 – Диаграмма классов

Пакет «Model» содержит так называемые роjo-классы (Plain Old Java Object), то есть простые Java-классы, не унаследованные от какого-то специфического класса и не реализующие никакие служебных интерфейсы сверх тех, которые нужны для бизнес-модели. Это классы:

- Publication (класс публикации);
- User (класс пользователя);
- Log (класс логов, не представлен на диаграмме).

Пакет «Beans» содержит так называемые JavaBeans – классы в языке Java, написанные по определённым правилам. В данном программном решении они используются для хранения отображаемых пользователю данных.

Пакет «Helpers» содержит вспомогательные классы, каждый из которых служит для определенных целей:

- DbHelper предназначен для работы с базой данных, выполняет операции поиска, извлечение и вставки информации. В связи с тем, что все операции, специфичные для работы с базой данных вынесены в один этот класс, возможный переход на другую СУБД не будет проблемным. В этом случае нужно будет реализовать всего один класс, работающий с новой СУБД.

- SparqlHelper служит для работы с точками доступа SPARQL и, как уже было сказано выше, использует библиотеку Jena. Чтобы вызывать удаленные

запросы существуют отдельные файлы для каждой точки доступа, так как у них используются разные пространства имен. Для подстановки ключевых слов в запрос используется компилирующий обработчик шаблонов (шаблонизатор) Freemarker<sup>5</sup>. Он является свободным программным обеспечением и позволяет отделить логику и данные от представления в рамках концепции Model-view-controller, а также используется для других, более специфичных задач, как в данном случае.

– **OntologyHelper** служит для преобразования описания публикаций из разных источников в описания, использующие общепринятые пространства имен, которые были выбраны ранее, а также для добавления результатов поискового запроса в локальную онтологию.

Пакет API содержит классы, необходимые для построения программного интерфейса, описанного далее.

### **3.4 Разработка программного интерфейса**

Для возможности включения разработанной системы в качестве вспомогательного модуля в другие программные решения было решено предоставить программный интерфейс приложения (API) для следующих функций:

- просматривать последние запросы и их результаты;
- искать по ключевому слову публикации из LOD-источников;
- выполнять SPARQL-запросы к локальной онтологии, построенной на основе предыдущих запросов пользователей.

Программный интерфейс реализуется посредством REST. REST (Representational state transfer) – это стиль архитектуры программного обеспечения для распределенных систем, таких как World Wide Web, который, как правило, используется для построения веб-служб.

---

<sup>5</sup> FreeMaker Java Template Engine. URL: <https://freemarker.apache.org/> (дата обращения 10.05.2018)

Для предоставления программного интерфейса был реализован веб-сервис с помощью фреймворка Jersey<sup>6</sup>. Jersey – Java-фреймворк с открытым исходным кодом, который был создан чтобы упростить разработку веб-сервисов, поддерживающих REST (RESTful веб сервисов) и их клиентов. Jersey предоставляет свой собственный API, а также многочисленные расширения SPI, чтобы разработчики могли расширить возможности Jersey.

Оформление документации к программному интерфейсу производилось в формате RAML 1.0<sup>7</sup>. RAML (RESTful API Modeling Language — язык для моделирования REST API) – это язык, предназначенный исключительно для описания API.

Преимуществами RAML являются:

- простой и логичный синтаксис, основанный на формате YAML<sup>8</sup>;
- поддержка наследования и возможность подключения внешних файлов спецификаций.

Дополнительным плюсом является наличие большого количества конвертеров, парсеров и генераторов интерактивной документации.

Описание операции получения результатов последних запросов в формате RAML представлено на рисунке 15. Результаты выводятся в формате JSON. Запрашивать их можно как через браузер, так и через специализированные приложения, такие как curl<sup>9</sup>. Выводятся следующие поля:

- ключевое слово, по которому производился запрос;
- дата запроса;
- источники, которые были выбраны для запроса;
- список публикаций с соответствующими полями, представленными ниже.

---

<sup>6</sup> Jersey. URL: <https://jersey.github.io/> (дата обращения 11.05.2018).

<sup>7</sup> RAML. URL: <http://raml.org/> (дата обращения 04.05.2018).

<sup>8</sup> YAML. URL: <http://yaml.org/> (дата обращения 04.05.2018).

<sup>9</sup> CURL. URL: <https://curl.haxx.se/> (дата обращения 05.05.2018).

```

##RAML 1.0
title: Semantic Search API
baseUri: https://semanticsearchtest.herokuapp.com/webresources/
/log:
get:
  description: Retrieve logs
  responses:
    200:
      body:
        application/json:
          example: |
            {
              "data": {
                "_id": { "$oid" : "5ad60569b93dbe216e863f74" },
                "className": "com.gerasin.oleg.semanticsearch.model.Log",
                "keyword": "semantic",
                "date": { "$date" : 1523975529344 },
                "sources": ["Open University", "Aalto University", "Springer"],
                "publications": [{ "Source" : "Aalto ..", "Title" : "Contextual ..", "Authors" : "http://.", "Date" : "2008" }],
              },
              "success": true,
              "status": 200
            }

```

Рисунок 15 – Описание программного интерфейса  
для получения результатов прошлых запросов

Описание поиска публикаций через программный интерфейс представлено на рисунке 16. Из него следует, что для поиска публикаций по ключевому слову “semantic” необходимо сделать запрос к пути `webresources/publications?keyword=semantic`.

Результатами запроса является список публикаций в формате JSON со следующими полями:

- источник;
- идентификатор;
- заголовок;
- список авторов;
- дата;
- описание.

У запроса есть два параметра:

- `keyword` – ключевое слово, по которому будет производиться поиск; обязательный; тип – текстовый;
- `remote` – флаг, при положительном значении выполняет удаленные запросы, при отрицательном – запросы к локальной онтологии; необязательный параметр; логический тип.

```

##RAML 1.0
title: Semantic Search API
baseUri: https://semanticsearchtest.herokuapp.com/webresources/
/publications:
  get:
    description: Retrieve publications
    queryParameters:
      keyword:
        displayName: Keyword
        type: string
        description: The system will search for that word in a set of publications
        example: semantic
        required: true
      remote:
        displayName: Remote
        type: boolean
        description: A flag that shows should system process remote query
        example: false
        required: false
        default: true
    responses:
      200:
        body:
          application/json:
            example: |
              {
                "data": {
                  "publications": [
                    {
                      "Source" : "Aalto ..",
                      "Title" : "Contextual ..",
                      "Authors" : "http://..", "Date" : "2008"
                    }
                  ],
                  "success": true,
                  "status": 200
                }
              }

```

Рисунок 16 – Описание программного интерфейса  
для получения публикаций по ключевому слову

Для выполнения запроса к онтологии, построенной на результатах прошлых запросов можно выполнить SPARQL-запрос к пути `webresources/sparql`, как показано на рисунке 17.

Принимает единственный параметр – текст SPARQL-запроса. Нет необходимости передавать префиксы, объявляющие пространства имен в тексте запроса, так как работая с локальной онтологией, используются общепринятые словари, а список доступных свойств показан в таблице 7.

```

#%RAML 1.0
title: Semantic Search API
baseUri: https://semanticsearchtest.herokuapp.com/webresources/
/sparql:
  get:
    description: Process sparql query
    queryParameters:
      query:
        displayName: Sparql query
        type: string
        description: Query to process
        example: SELECT distinct ?Title ?Authors ?Date WHERE { [] a bibo:Article ; ...
        required: true
    responses:
      200:
        body:
          application/json:
            example: |
              {
                "data": {
                  "head": {
                    "vars": [ "Title" , "Authors" , "Date" ]
                  } ,
                  "results": {
                    "bindings": [
                      {
                        "Title": { "type": "literal" , "value": "On the Evolution of the Word and the Concept." } ,
                        "Authors": { "type": "uri" , "value": "http://data.aalto.fi/id/publication/rescat/rec8438\_author1" } ,
                        "Date": { "type": "literal" , "value": "2010" }
                      }
                    ]
                  }
                } ,
                "success": true,
                "status": 200
              }

```

Рисунок 17 – Описание программного интерфейса  
для получения результатов SPARQL-запросов



## 4 ЭКСПЕРИМЕНТАЛЬНОЕ ИССЛЕДОВАНИЕ ИНФОРМАЦИОННО-ПОИСКОВОЙ СИСТЕМЫ

### 4.1 Тестирование системы

Для проверки работоспособности системы были разработаны тесты с использованием unit тестирования и тестирования веб-интерфейса.

Для unit-тестирования использовался один из самых известных, а также один из самых используемых фреймворк для тестирования для языка Java – JUnit<sup>10</sup> 5. Пример теста показан на рисунке 18.

```
@Test
public void cachedPublicationsSizeTest()
{
    DbHelper dbHelper = new DbHelper();

    cachedPublications = dbHelper.getCachedPublications(
        "semantic",
        Arrays.asList(new String[]{
            SparqlHelper.AALTO,
            SparqlHelper.OU
        }));

    assertEquals(12, cachedPublications.size());
}
```

Рисунок 18 – Пример теста с использованием JUnit

В общей сложности для тестирования серверного кода было написано более 30 unit-тестов. Все тесты успешно были пройдены в процессе тестирования.

Для тестирования клиента была использована программная библиотека для управления браузерами Selenium WebDriver, которая входит в проект Selenium. Для настройки работы с системой используется ChromeDriver, как изображено на рисунке 19.

---

<sup>10</sup> JUnit. URL: <https://junit.org/junit5/> (дата обращения 11.05.2018).

```
private static WebDriver driver;

@BeforeClass
public static void setup()
{
    System.setProperty("webdriver.chrome.driver", "/home/user/Work/chromedriver");
    driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    driver.get("https://semanticsearchtest.herokuapp.com/login.xhtml");
}
```

Рисунок 19 – Первичная настройка драйвера для работы с браузером

Пример тестирования с помощью библиотеки Selenium WebDriver представлен на рисунке 20.

```
@Test
public void userLogin()
{
    WebElement loginField = driver.findElement(By.id("username"));
    loginField.sendKeys("geras");
    WebElement passwordField = driver.findElement(By.id("password"));
    passwordField.sendKeys("123");
    WebElement loginButton = driver.findElement(By.xpath("//button[text()='Login']"));
    loginButton.click();
    WebElement profileUser = driver.findElement(By.cssSelector(".login-button__user"));
    String username = profileUser.getText();
    Assert.assertEquals("geras", username);
}

@AfterClass
public static void tearDown()
{
    WebElement logoutButton = driver.findElement(By.id("login__logout"));
    logoutButton.click();
    driver.quit();
}
```

Рисунок 20 – Пример тестирования клиента

Всего для тестирования веб-интерфейса было написано более 20 тестов. В процессе тестирования все они были пройдены успешно.

## 4.2 Пример решения задачи

Рассмотрим пример использования системы для поиска публикаций.

Изначально пользователю необходимо зарегистрироваться в системе. Для этого вводится желаемый логин и пароль, по которым будет производиться дальнейшая аутентификация пользователя, как показано на рисунке 21.

Semantic search x

Secure | https://semanticsearchtest.herokuapp.com/login.xhtml

Username:

Password:

Login Sign up

Рисунок 21 – Страница регистрации и входа в систему

На рисунке 22 изображена форма ввода ключевых слов, которые будут сохранены как интересы пользователя в системе и будут использованы в дальнейшем для персонализированного подбора публикаций.

geras Logout

Your interests: lod x linked data x semantic web x semantic search x

Save Cancel

Рисунок 22 – Страница с интересами пользователя

На самой странице поиска пользователь вводит ключевые слова, по которым он хочет найти публикации и выбирает источники данных – открытые точки доступа, а также устанавливает флаг – будут ли результаты для запроса браться из локальной онтологии, как показано на рисунке 23.

geras Logout

Your query: semantic

Sources: Springer x Open University x Aalto University x Publications Europe x

Search

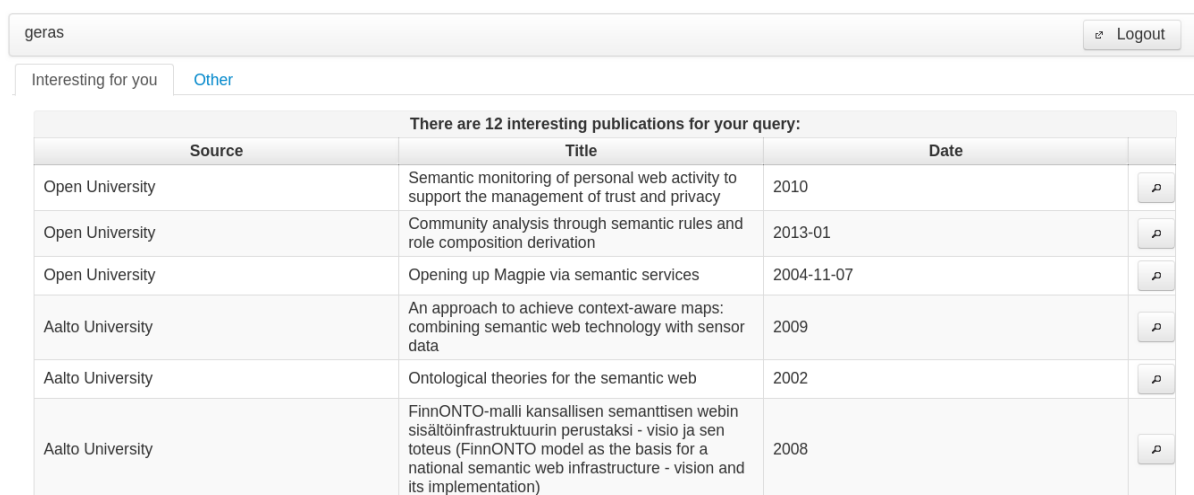
Do you want to search in cached publications for your query if it's possible? That's much faster. ☒

Рисунок 23 – Поисковая страница

Если пользователь установил флаг, то система ищет в базе данных и локальной онтологии был ли такой запрос ранее. Если соответствующие записи находятся, то результаты для поискового запроса берутся из неё.

Если запись не была найдена, или пользователь не пожелал брать кешированные результаты, то система делает удалённые SPARQL-запросы к выбранным пользователем точкам доступа. После этого осуществляется запись результатов в базу данных и локальную онтологию, причем с использованием общепринятых словарей (пространств имен), которые были выбраны в главе 2.

На рисунке 24 представлена страница с результатами. Сверху страницы расположены вкладки, изначально выбрана вкладка с результатами, которые могут быть интересны пользователям. Система определяет, интересны ли результаты поиска по ключевым словам, которые пользователь ввел в поле “интересы”. Есть возможность перейти ко всем результатам, а также посмотреть детальную информацию по каждой публикации. Особая ценность заключается в том, что, так как это связанные открытые данные, пользователь может перейти по URI идентификатору публикации или автора, и посмотреть все связанные сущности.









There are 12 interesting publications for your query:			
Source	Title	Date	
Open University	Semantic monitoring of personal web activity to support the management of trust and privacy	2010	
Open University	Community analysis through semantic rules and role composition derivation	2013-01	
Open University	Opening up Magpie via semantic services	2004-11-07	
Aalto University	An approach to achieve context-aware maps: combining semantic web technology with sensor data	2009	
Aalto University	Ontological theories for the semantic web	2002	
Aalto University	FinnONTO-malli kansallisen semanttisen webin sisältöinfrastruktuurin perustaksi - visio ja sen toteutus (FinnONTO model as the basis for a national semantic web infrastructure - vision and its implementation)	2008	

Рисунок 24 – Страница с результатами

### 4.3 Научная и практическая значимость работы

Разработанная система работает со связанными открытыми данными высокого качества, так как их предоставляют университеты и другие надежные организации.

До разработки данной системы для поиска публикаций среди используемых источников связанных открытых данных, было необходимо обращаться отдельно к каждой точке доступа, подбирать индивидуальные SPARQL-запросы, что, конечно же, неудобно. Данная работа призвана решить данную проблему, предоставляя единую информационно-поисковую систему, осуществляющую удаленные запросы централизованно.

Особая ценность работы со связанными открытыми данными состоит в том, что при их публикации указываются ссылки на связанные ресурсы, что позволяет пользователям переходить по ним, открывая новые факты. Но что еще важнее, что не только пользователь может воспользоваться данной возможностью, но и машина может использовать связи между данными и получать новые знания, которые, возможно, при публикации были не так очевидны.

Система не только представляет единую точку доступа к публикациям других источников, но и выбирает на основании предоставленной пользователем информации те записи, которые должны его заинтересовать. Данная особенность позволяет сократить время поиска нужных публикаций, так как самые релевантные с точки зрения интересов пользователя, выводятся отдельно от остальных.

К сожалению, данные из разных источников используют разные словари (пространства имен), что затрудняет вывод новых знаний на их основе. Разработанная система призвана решить данную проблему, формируя собственную онтологию на основе запросов, используя общепринятые пространства имен, что позволяет формировать больше знаний о предметной области.

Для последующей обработки данных реализован REST API, который позволяет:

- просматривать последние запросы и их результаты;
- искать по ключевому слову публикации из LOD-источников;
- выполнять SPARQL-запросы к локальной онтологии, построенной на основе предыдущих запросов пользователей.

С помощью предоставляемого программного интерфейса возможно использование разработанной системы как вспомогательного модуля в других системах. Для развертывания системы на локальных машинах создан Docker-образ<sup>11</sup>, а для централизованного доступа к прототипу и базе данных приложение развернуто на Heroku<sup>12</sup>.

---

<sup>11</sup> Docker Hub. URL: <https://hub.docker.com/r/ovger/semanticsearchtest/> (дата обращения 24.02.2018).

<sup>12</sup> Semantic Search. URL: <https://semanticsearchtest.herokuapp.com/> (дата обращения 24.02.2018).

## ЗАКЛЮЧЕНИЕ

В ходе выполнения работы были решены следующие задачи:

- проведен анализ предметной области (связанных открытых данных), выявлены существующие проблемы в исследуемой области. К основным проблемам можно отнести следующие: отсутствие современных инструментальных средств для централизованного поиска между источниками LOD, относительная сложность языка запросов SPARQL, отсутствие возможности накопления и повторного использования экспертных знаний при решении аналогичных задач;
- предложен новый подход к решению задачи сбора информации о публикациях из источников связанных открытых данных и дальнейшей их обработки. Предложенный подход предполагает централизованный поиск по источникам LOD, посредством удаленных запросов к точкам доступа;
- разработана структура онтологической модели представления знаний предметной области, необходимая для хранения результатов прошлых запросов, обеспечивающая поддержку логического вывода новых знаний на основе существующих;
- реализована система, обеспечивающая решение задачи сбора информации о публикациях из источников связанных открытых данных. Что немаловажно, система выводит найденные публикации, учитывая интересы пользователя. Данные о найденных публикациях преобразуются в описания, использующие стандартные пространства имен, и заносятся в новую онтологию, за счет чего образуется более полная картина представления знаний предметной области;
- разработанные модели и система протестированы, полученный результат позволяет сделать вывод об их эффективности.

Таким образом, поставленные в работе цели были достигнуты, а задачи решены. Совокупность решенных задач позволяет осуществлять поиск среди связанных открытых данных, скрывая сложность запросов для конечного пользователя, что может повысить популярность и стимулировать развитие этого

перспективного направления. Разработанная система предоставляет обширные возможности по дальнейшей обработке полученных результатов за счет предоставления документированного программного интерфейса.

В качестве перспективных работ можно выделить следующие: автоматическое определение интересов пользователя на основе его предыдущих запросов или публикаций, увеличение числа источников связанных открытых данных, расширение возможностей поисковых запросов.



## СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Wu, X., Zhu, X., Wu, G., Ding, W. Data mining with big data. // IEEE Transactions on Knowledge and Data Engineering. 2014. V. 26. P. 97-107.
2. Bizer, C., Heath, T. Linked Data: Evolving the Web into a Global Data Space. // Synthesis Lectures on the Semantic Web. 2011. V. 1. N. 1. P. 1-136.
3. Муромцев Д.И., Леманн Й., Семерханов И.А., Навроцкий М.А., Ермилов И.С. Исследование актуальных способов публикации открытых научных данных в сети // Научно-технический вестник информационных технологий, механики и оптики. 2015. Том 15. № 6. С. 1081–1087.
4. Zablith F., Fernandez M., Rowe M. The OU Linked Open Data: Production and Consumption. // The Semantic Web: ESWC 2011 Workshops. 2011. P. 36-39
5. SPARQL 1.1 Query Language [Электронный ресурс]. — Режим доступа: <https://www.w3.org/TR/sparql11-query/>, свободный. Яз. англ. (дата обращения 12.12.2017).
6. Auer S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., & Ives, Z. Dbpedia: A nucleus for a web of open data // The semantic web. Springer, Berlin, Heidelberg, 2007. P. 722-735.
7. Jessop D. M., Adams S. E., Murray-Rust P. Mining chemical information from open patents // Journal of cheminformatics. 2011. V. 3. N. 1. P. 40.
8. K. Braunschweig, J. Eberius, M. Thiele and W. Lehner, The State of Open Data. Limits of Current Open Data Platforms. // World Wide Web Conference, Lyon, France, 2012.
9. European Union Open Data Portal [Электронный ресурс]. — Режим доступа: <http://data.europa.eu/euodp/en/data/>, свободный. Яз. англ. (дата обращения 20.01.2018).
10. Junar Data Platform [Электронный ресурс]. — Режим доступа: <http://www.junar.com/>, свободный. Яз. англ. (дата обращения 02.02.2018).

11. Eurostat [Электронный ресурс]. — Режим доступа: <http://ec.europa.eu/eurostat/web/main>, свободный. Яз. англ. (дата обращения 05.02.2018).

12. Principles P. Principles for open data in science [Электронный ресурс]. — Режим доступа: <http://pantonprinciples.org>, свободный. Яз. англ. (дата обращения 15.02.2018).

13. Conformant Licenses [Электронный ресурс]. — Режим доступа: <https://opendefinition.org/licenses/>, свободный. Яз. англ. (дата обращения 27.02.2018).

14. Bizer C., Heath T., Berners-Lee T. Linked data-the story so far // International journal on semantic web and information systems. 2009. V. 5. N. 3. P. 1-22.

15. Linked data [Электронный ресурс]. — Режим доступа: <https://www.w3.org/standards/semanticweb/data>, свободный. Яз. англ. (дата обращения 20.02.2018).

16. Linked Data - Design Issues [Электронный ресурс]. — Режим доступа: <https://www.w3.org/DesignIssues/LinkedData.html>, свободный. Яз. англ. (дата обращения 25.02.2018).

17. Kauppinen, T., & Espindola, G.M. Linked Open Science-Communicating, Sharing and Evaluating Data, Methods and Results for Executable Papers. // The International Conference on Computational Science (ICCS 2011). Elsevier Procedia Computer Science series, Singapore, June, 2011.

18. Радченко И.А. Использование открытых данных в научных исследованиях. // Информационное общество 2013. №1-2. С. 93-101/

19. Linked Open Aalto Data Service [Электронный ресурс]. - Режим доступа: <http://data.aalto.fi/>, свободный. Яз. англ. (дата обращения 23.04.2018).

20. Linked Data from The Open University [Электронный ресурс]. - Режим доступа: <http://data.open.ac.uk/>, свободный. Яз. англ. (дата обращения 02.05.2018).

21. Creative Commons [Электронный ресурс]. - Режим доступа: <https://creativecommons.org/licenses/by/3.0/>, свободный. Яз. англ. (дата обращения 01.05.2018).

22. EU Open Data Portal [Электронный ресурс]. - Режим доступа: <http://data.europa.eu/euodp/en/home/>, свободный. Яз. англ. (дата обращения 01.05.2018).

23. Springer Nature SciGraph Data Explorer [Электронный ресурс]. - Режим доступа: <http://scigraph.springernature.com/explorer/>, свободный. Яз. англ. (дата обращения 01.05.2018).

24. Bibliographic Ontology Specification [Электронный ресурс]. - Режим доступа: <http://bibliontology.com/>, свободный. Яз. англ. (дата обращения 24.04.2018).

25. FOAF Vocabulary Specification [Электронный ресурс]. - Режим доступа: <http://xmlns.com/foaf/spec/>, свободный. Яз. англ. (дата обращения 24.04.2018).

26. Dublin Core Metadata Initiative [Электронный ресурс]. - Режим доступа: <http://dublincore.org/documents/dcmi-terms/>, свободный. Яз. англ. (дата обращения 24.04.2018).

27. Protege [Электронный ресурс]. - Режим доступа: <https://protege.stanford.edu/>, свободный. Яз. англ. (дата обращения 03.05.2018).