Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт Кибернетики
Направление подготовки 09.04.01 Информатика и вычислительная техника
Кафедра Оптимизации систем управления

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Тема работы
Интеграция баз данных используя технологию Semantic Web

УДК 004.658.6

Студент

Группа	ФИО	Подпись	Дата
8ВМ4В	Рафиков Марк Рамильевич		

Руководитель

Должность	ФИО	Ученая степень, звание	Подпись	Дата
профессор каф. ОСУ	Тузовский Анатолий Федорович	д.т.н.		

КОНСУЛЬТАНТЫ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
доцент каф. МЕН	Конотопский В.Ю.	к.э.н.		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент	Акулов П.А.			

ДОПУСТИТЬ К ЗАЩИТЕ:

Зав. кафедрой	ФИО	Ученая степень, звание	Подпись	Дата
ОСУ	Иванов М.А.	к.т.н.		

Томск – 2016 г.

ЗАПЛАНИРОВАННЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ ПО ПРОГРАММЕ

Код результатов	Результат обучения (выпускник должен быть готов)
Общепрофессиональные компетенции	
Р1	Воспринимать и самостоятельно приобретать, развивать и применять математические, естественнонаучные, социально-экономические и профессиональные знания для решения нестандартных задач, в том числе в новой или незнакомой среде и в междисциплинарном контексте.
Р2	Владеть и применять методы и средства получения, хранения, переработки и трансляции информации посредством современных компьютерных технологий, в том числе в глобальных компьютерных сетях.
Р3	Демонстрировать культуру мышления, способность выстраивать логику рассуждений и высказываний, основанных на интерпретации данных, интегрированных из разных областей науки и техники, выносить суждения на основании неполных данных, анализировать профессиональную информацию, выделять в ней главное, структурировать, оформлять и представлять в виде аналитических обзоров с обоснованными выводами и рекомендациями.
Р4	Анализировать и оценивать уровни своих компетенций в сочетании со способностью и готовностью к саморегулированию дальнейшего образования и профессиональной мобильности. Владеть, по крайней мере, одним из иностранных языков на уровне социального и профессионального общения, применять специальную лексику и профессиональную терминологию языка.
Профессиональные компетенции	
Р5	Выполнять инновационные инженерные проекты по разработке аппаратных и программных средств автоматизированных систем различного назначения с использованием современных методов проектирования, систем автоматизированного проектирования, передового опыта разработки конкурентно способных изделий.
Р6	Планировать и проводить теоретические и экспериментальные исследования в области проектирования аппаратных и программных средств автоматизированных систем с использованием новейших достижений науки и техники, передового отечественного и зарубежного опыта. Критически оценивать полученные данные и делать выводы.
Р7	Осуществлять авторское сопровождение процессов проектирования, внедрения и эксплуатации аппаратных и программных средств автоматизированных систем различного назначения.
Общекультурные компетенции	
Р8	Использовать на практике умения и навыки в организации исследовательских, проектных работ и профессиональной эксплуатации современного оборудования и приборов, в управлении коллективом.
Р9	Осуществлять коммуникации в профессиональной среде и в обществе в целом, активно владеть иностранным языком, разрабатывать документацию, презентовать и защищать результаты инновационной инженерной деятельности, в том числе на иностранном языке.
Р10	Совершенствовать и развивать свой интеллектуальный и общекультурный уровень. Проявлять инициативу, в том числе в ситуациях риска, брать на себя всю полноту ответственности.
Р11	Демонстрировать способность к самостоятельному обучению новым методам исследования, к изменению научного и научно-производственного профиля своей профессиональной деятельности, способность самостоятельно

Код результатов	Результат обучения (выпускник должен быть готов)
	приобретать с помощью информационных технологий и использовать в практической деятельности новые знания и умения, в том числе в новых областях знаний, непосредственно не связанных со сферой деятельности, способность к педагогической деятельности.

Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт Кибернетики
Направление подготовки 09.04.01 Информатика и вычислительная техника
Кафедра Оптимизации систем управления

УТВЕРЖДАЮ:
Зав. кафедрой
_____ _____ Иванов М.А.
(Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

Магистерская диссертация

(бакалаврской работы, дипломного проекта/работы, магистерской диссертации)

Студенту:

Группа	**ФИО**
8ВМ4В	Рафиков Марк Рамильевич

Тема работы:

Интеграция баз данных используя технологию Semantic Web	
Утверждена приказом директора (дата, номер)	19.02.2016 №1321/с

Срок сдачи студентом выполненной работы:	06.06.2016

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе *(наименование объекта исследования или проектирования; производительность или нагрузка; режим работы (непрерывный, периодический, циклический и т. д.); вид сырья или материал изделия; требования к продукту, изделию или процессу; особые требования к особенностям функционирования (эксплуатации) объекта или изделия в плане безопасности эксплуатации, влияния на окружающую среду, энергозатратам; экономический анализ и т. д.).*	

Перечень подлежащих исследованию, проектированию и разработке вопросов *(аналитический обзор по литературным источникам с целью выяснения достижений мировой науки техники в рассматриваемой области; постановка задачи исследования, проектирования, конструирования; содержание процедуры исследования, проектирования, конструирования; обсуждение результатов выполненной работы; наименование дополнительных разделов, подлежащих разработке; заключение по работе).*	
Перечень графического материала *(с точным указанием обязательных чертежей)*	

Консультанты по разделам выпускной квалификационной работы

(с указанием разделов)

Раздел	Консультант
Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	Конотопский В.Ю.
Социальная ответственность	Акулов П.А.
Раздел ВКР, выполненный на иностранном языке	Куркан Н.В.

Названия разделов, которые должны быть написаны на русском и иностранном языках:

Обзор литературы
Объект и методы исследования
Разработанное решение
Программная реализация системы
Финансовый менеджмент, ресурсоэффективность и ресур- сосбережение
Социальная ответственность
The proposed solution

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	15.06.2015

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
профессор каф. ОСУ	Тузовский Анатолий Федорович	д.т.н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ВМ4В	Рафиков Марк Рамильевич		

Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт Кибернетики
Направление подготовки (специальность) 09.04.01 Информатика и вычислительная техника
Уровень образования – бакалавр
Кафедра Оптимизации система управления
Период выполнения – осенний / весенний семестр 2015/2016 учебного года

Форма представления работы:

Магистерская диссертация

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Срок сдачи студентом выполненной работы:	25.05.2016

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
10.03.2015	*Обзор литературы*	*10*
5.04.2015	*Объект и методы исследования*	*20*
20.04.2014	*Разработанное решение*	*20*
7.05.2015	*Программная реализация информационной системы*	*20*
15.05.2015	*Финансовый менеджмент, ресурсоэффективность и ресур- сосбережение*	*10*
18.05.2015	*Социальная ответственность*	*10*
21.05.2015	*The proposed solution*	*10*

Составил преподаватель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
профессор каф. ОСУ	Тузовский А.Ф	д.т.н.		

СОГЛАСОВАНО:

Зав. кафедрой	ФИО	Ученая степень, звание	Подпись	Дата
зав. каф. ОСУ	Иванов М.А.	к.т.н.		

Реферат

Выпускная квалификационная работа 100 с, 20 рис, 16 табл, 47 источников, 2 прил.

Ключевые слова: Semantic Web, RDF, интеграция, базы данных.

Цель работы - разработка и исследование методов и моделей интеграции реляционных баз данных с использованием технологий и инструментов Semantic Web.

Объектом исследования являются разнородная информация, хранящаяся в реляционных баз данных, а также методы и технологии Semantic Web.

В процессе исследования проводилось изучение возможностей платформы D2RQ и программной библиотеки RDFLib

В результате исследования были разработаны архитектура приложения и веб-сервис для интегрирования реляционных баз данных.

Областью применения является интеграция информации из нескольких реляционных баз данных

В будущем планируется развертывание и публикация веб сервиса в сети Интернет

Оглавление

Введение

На многих крупных предприятиях информация хранится в различных базах данных. Информация этих баз данных может повторяться или дополнять друг друга. Работа с распределёнными источниками баз данных сложный и не удобный процесс. Чаще всего, разные базы данных имеют различные схемы. Это не позволяет интегрировать информацию в одну большую базу данных, содержащую необходимые для работы таблицы.

Существует множество решений, относительно способов хранения информации, содержащейся в базах данных. Но эти решения не избавляют от проблемы интеграции данных из нескольких источников. Интеграция информации реляционных баз данных является актуальной задачей для специалистов, связанных с обработкой данных из разнородных источников.

В тоже время для решения задачи работы с разнородными данными WWW организация Всемирной Паутины развивает идею создания следующей версии WWW – Семантической Паутины (Semantic Web). Семантическая Паутина предполагает хранение все информации Всемирной Паутины в формате понятном и вычислительной машине, и человеку. Кроме этого данный формат предоставляет возможность выполнять на этих данных логических заключений. Семантическая паутина существует сейчас как надстройка над Всемирной Паутиной, но, к сожалению, глобально не используется. Одной из основных проблем внедрения Семантической Паутины является перевод больших объемов уже существующей информации из реляционных баз данных в формат данных Семантической паутины.

Для реализации SW уже разработано большое количество стандартов: (например, RDF/RDFS, OWL, SPARQL) и различных программных инструментов (например, D2R и Protégé).

Основываясь на идеях и стандартах SW можно исследовать возможность интеграции реляционных БД организации.

Цель работы: разработать и исследовать методы и модели интеграции реляционных баз данных с использованием технологий и инструментов Semantic Web.

Объектом исследования является разнородная информация, хранящаяся в реляционных баз данных, а также методы и технологии Semantic Web.

Практическая новизна заключается в возможности интеграции баз данных в формат понятный для машинной. Это исследование может стать прочным основанием глобального использования поисковых систем, основанных только на формализованных данных. Данное исследование, также может позволить сэкономить большим организациям, которые имеют огромные базы данных

Результаты данной работы были доложены на конференции Молодежь и современные информационные технологии [1]

1. Обзор литературы (аналитический обзор)

Разработка систем управления базами данных в 90-х годах сопровождалась развитием технологий создания широкомасштабных (wide-area) компьютерных сетей. Стало само собой очевидным, что хранение и поиск, а следовательно и интеграция распределенной информации становится все более и более важным для предприятий и распределенных (децентрализованных) организаций. С непрерывным ростом распространения технологий глобальных сетей (WAN) и Интернета, важность такой интеграции стала все более возрастать. Стало очевидным, что традиционные подходы, разработанные для систем мультибаз данных и распределенных баз данных стали уже не подходящими (адекватными) в особенности в тех случаях, когда интегрируемая информация поддерживалась в очень автономных и распределенных системах. Хотя многие базовые алгоритмы являются сходными, для того, чтобы они могли решать новые проблемы они должны быть расширены и объединены с новыми понятиями. Для интеграции информации обычно применяется подход посредников-адаптеров, если информация предоставляется различными источниками, такими, как разные виды систем баз данных (СУБД), наборами XML документов и или web-сайтами и разными типами интерфейсов, таких, как SQL, XQuery или Web сервисами (services).

1.1. Посредники в будущих информационных системах

Один из сторонников подхода на основе посредников (mediator approach), Gio Wiederhold следующим образом сформулировал понятие посредника [2]:

“Посредник (mediator) это программный модуль, который использует закодированные знания о некоторых множествах или подмножествах данных для того, чтобы создавать информацию для приложений более высокого уровня”.

Данная цитата может использоваться как высокоуровневое определение архитектуры, основанной на посредниках. В соответствии с [2], сообщество исследователей столкнулось с двумя типами проблем:

1. “для отдельных баз данных основным препятствием для доступа конечных пользователей является объем данных, которые становятся доступными, недостаток абстракции и потребность понять представление данных”
2. “большой проблемой при объединении информации из множества баз данных является несоответствие, которые встречаются в представлении и структуре информации”

Проблемы первого вида были хорошо решены в течении последней пары десятилетий: сегодня, не является серьезной проблемой управление и поиск информации в больших объемах данных. OLAP технологии являются достаточно мощными, чтобы было возможным абстрагироваться от подробно детализированных данных и очень быстро предоставить более значимую (more meaningful) информацию. Однако до сих пор еще осталось пролемой правильная интерпретация (понимание) данные представленные в базах данных.

Второй тип проблем, говоря кратко, является основной целью текущих исследований в области Semantic Web. При объединении (т.е. интеграции) информации из различных источников обычной проблемой является правильное согласование их семантик, в особенности, если базовые данные должны объединяться и обрабатываться автоматически. Wiederhold перечислил некоторые из проблем, которые встречаются при объединении автономных источников, к которым относятся:

- конфликты имен,
- разная детальность уровня абстрагирования ,
- разный временной контекст,
- различие в семантике предметных областей,
- различие в семантике значений, и т.п.

Более того, он описал абстрактную модель обработки информации, ссылаясь на статью [3] «Как мы можем думать» (“As We May Think”). Wiederhold назвал процесс получения, обработки и коммуникации – информационным планированием, так как данные берутся из прошлого с целью оказания воздействия на будущее. Он утверждает, что интерфейсы будущих информационных систем должны брать на себя активную роль в поддержке пользователей в ходе такого планирования. Он называет посредник «модулем, который занимает явный, активный слой между приложениями пользователей и источниками данных» и описывает их архитектуру так, как это показано на рисунке 1.1.

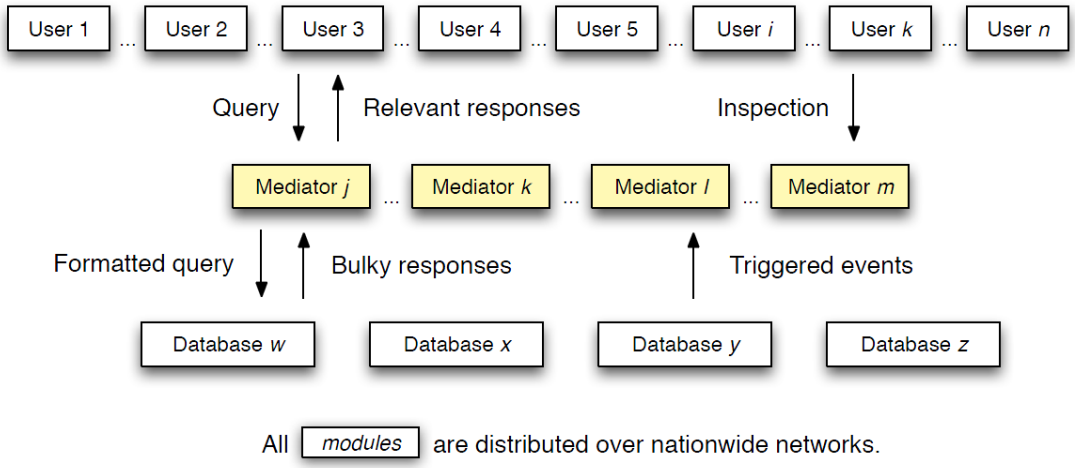


Рисунок 1.1 – Посредники в качестве интерфейсов для потоков информации [2]

Такое понимание посредников похоже на теорию программных агентов (software agents), которые действуют от имени своих пользователей и выполняют различные задачи. Например, целями систем Carnot [4] и InfoSleuth [Bayardo et al. 1997] была разработка основанных на агентах обобщенных сред со специализированными программными агентами, ответственными за обнаружение ресурсов (resource discovery), организацию взаимодействия и интеграцию информации и сервисов в динамических средах. Подход, основанный на интеллектуальных программных агентах, выполняющих сложные задачи, которые обычно требуют некоторого подобия (вида) интеллекта человека, является фактически весьма амбициозным и трудным для реализации.

В контексте интеграции информации, посредники должны быть способными обрабатывать запросы путем анализа произвольного количества разных известных источников информации. Обычно это достигается путем размещения на каждом источнике данных адаптера, который является ответственным за обработку локального запроса и преобразование данных. В следующих разделах рассматриваются конкретные реализации таких, основанных на посредниках, систем интеграции информации. В основном, все подходы могут быть разделены на проекты, выполняемые в сообществе исследователей баз данных (как например, система Garlic компании IBM) и проекты, выполняемые в сообществах исследователей искусственного интеллекта и вычислений на основе агентов (как например, InfoSleuth).

Первым будет описан проект Garlic компании IBM, так как он уже включал многие важные особенности (аспекты) типичных систем, основанных на посредниках-адаптерах.

1.2. IBM “Garlic”

Технология IBM “Garlic”, использующая федерированные сервера для поиска необходимой информации об объекте из различных, автономных источников [6].

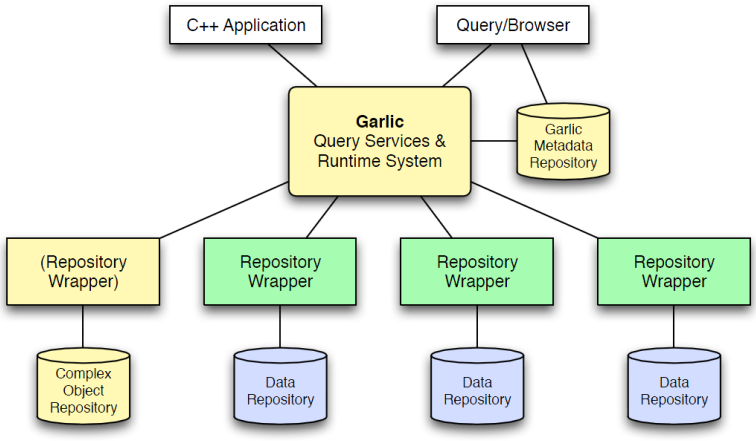


Рисунок 1.2 – Архитектура системы Garlic

В центральной части рисунке 1.2. показан посредник системы Garlic, включающий сервисы запросов и систему выполнения, а также хранилище метаданных. Система Garlic может быть связана с приложениями на языке C++

с помощью специального API. Интерактивный компонент обработки запросов/просмотра предоставляет конечным пользователям более удобный способ выполнения доступа к интегрированной информации. Этот компонент был новым подходом выполнения запросов, который назвался «запросы-по-графическому-шаблону». Фактически, данный подход был очень похож на появляющиеся гипермедиа браузеры. В сравнении с World Wide Web, который применял намного менее ограниченный и не скоординированный способ организации связей и ссылок информация в системе Garlic поступала из баз данных и была составлена на основе алгебраических правил.

В нижней части рисунке. 1.2 показано несколько хранилищ данных, к каждому из них прикреплен адаптер, связанный с системой Garlic. Работа адаптера состоит в переводе информации о типах и схемах данных и преобразовании запросов в родной для источника данных язык запросов или в выполнении API вызовов к удаленному хранилищу. Вся информация о глобальной схеме системы Garlic и другая информация, связанная с выполнением перевода, поддерживается в хранилище метаданных. Одним из таких хранилищ является специальное хранилище: Complex Object Repository (Хранилище Сложных Объектов), которое используется для общего связывания всей информации. Оно может располагаться в удаленном хранилище (repository) или в самой системе Garlic и использоваться для предоставления дополнительной информации для связывания родственных, но ранее никак не соединяемых между собой объектов из разных хранилищ.

Архитектура системы Garlic основывается на следующих предположениях:

- для любого вида источника данных (реляционных баз данных различных производителей, более старых, не реляционных баз данных, как например, IMS компании IBM, файловых систем, основанных на записях, мультимедиа баз данных со специфическими для мультимедиа возможностями поиска и т.п.) должна иметься возможность написать адаптер;

- адаптер должен использовать возможности базовой информационной системы
    - выбор на основе ключевых слов,
    - интервальные запросы ,
    - булевые выражения,
    - сопоставление на основе сходства
- должна быть совершенно простая возможность создать новый адаптер или расширить новыми возможностями существующие адаптеры (например, по мере развития возможностей удаленного выполнения работы (remote capabilities))
- добавление нового адаптера к данной архитектуре не должно влиять (не должно зависеть) на другие компоненты системы.

1.3. Проект TSIMMIS

Проект TSIMMIS (аббревиатура для **T**he **S**tanford-**I**BM **M**anager of **M**ultiple **I**nformation **S**ources) был совместным проектом исследовательского центра Almaden Research Center компании IBM и Stanford University в тоже самое время, когда в этом центре выполнялся проект Garlic[7]. Однако, архитектура системы TSIMMIS во многом отличается от архитектуры системы Garlic.

Во-первых, в архитектуре системы TSIMMIS используется упорядоченные в стеки посредники, каждый из которых принимает вход от множества адаптеров или других посредников и выполняет некоторую обработку, чтобы внести некоторую добавочную стоимость. В системе TSIMMIS, адаптеры называются трансляторами. Общая схема архитектуры TSIMMIS показана на рисунке. 1.3.

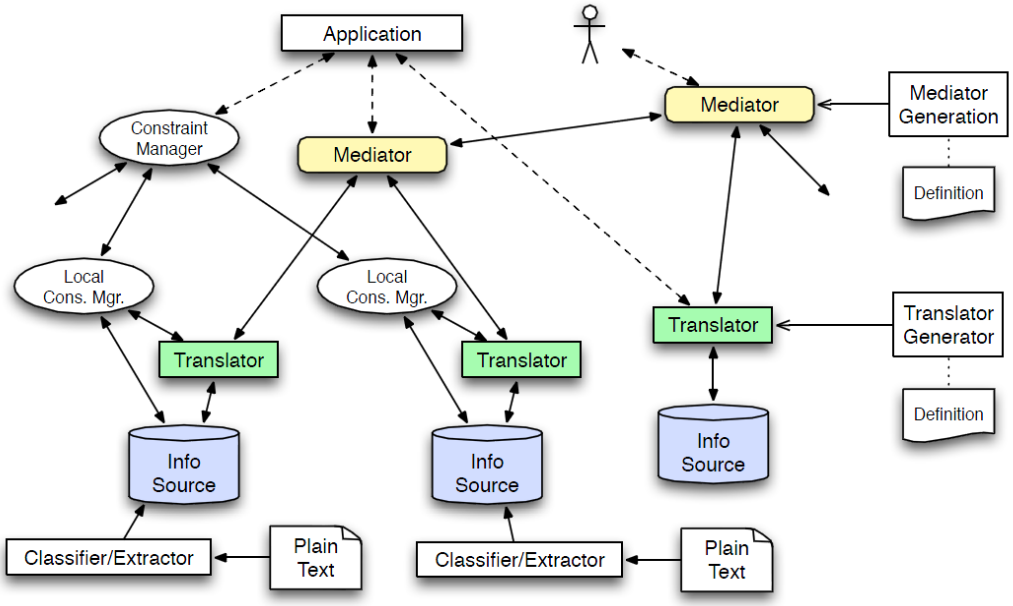


Рисунок 1.3 – Архитектура системы TSIMMIS.

Как показано на рисунке. 1.3, пользователи могут выбирать один из существующих посредников (а также напрямую адаптеры) для выполнения запросов. Посредники имеют фиксированное поведение, которое жестко кодируется, как часть их реализации. Например, посредники могут объединять два имеющих адаптеры источника данных с помощью таких операций, как join, union или других операций [7]. В то время, как один адаптер может предоставлять информацию о библиотеке, посредник может добавлять обложки книг, полученные от другого адаптера в промежуточные результаты. Другой посредник может включать расширенную информацию об авторе или отзывы пользователей на третьем шаге. Каждый интерфейс запросов является чем-то сходным с более или менее сложным специфическим представлением доступных элементов данных. Как будет показано далее, язык запросов в системе TSIMMIS явно не предоставляет операторы, которые имеются в реляционной алгебре и других декларативных языках запросов. Пользователь не имеет непосредственной возможности описать операции join или union, это возможно только в том случае, если нет посредника предоставляющего такое конкретное представление двух других источников, которые могут быть посредниками или адаптерами [8].

Подходе TSIMMIS очень четко соответствует видению Gio Wiederhold на архитектуру, основанную на посредниках-адаптерах, который был описан в разделе 1.1, хотя можно спорить, что эти возможности достаточно ограничены, так как связывания и другие типичные операции должны явно обеспечиваться посредниками. Потребность в большом наборе разных посредников также стало поводом для разработчиков системы TSIMMIS разрабатывать генераторы для автоматического создания, как адаптеров, так и посредников. Адаптеры и посредники могут генерироваться на основе правил и программных библиотек.

Во-вторых, адаптеры системы TSIMMIS были расширены сортировщиками и экстракторами, которые способны извлекать структурированные данные из не структурированных источников, таких, как простой текст или HTML документы. Система Garlic также поддерживает интеграцию мультимедиа источников данных, но в сравнении с TSIMMIS, в ней предполагается, что мультимедиа система предоставляет специальный адаптер, который предоставляет STAR и POP элементы, с помощью которых можно напрямую использовать специфические функции доступа. Система TSIMMIS предоставляет сортировщики и извлекатели независимо от адаптеров. Однако, такая особенность может также рассматриваться как расширение, которые являются совершенно независимыми от системы посредников-адаптеров. Источники мультимедиа данных также могут предоставлять собственные метаданные и в этом случае не требуется специальных экстракторов. В отличие от подхода системы Garlic к интеграции мультимедия источников, сортировщики и экстракторы TSIMMIS могут прикрепляться к адаптерам любой другой системы интеграции и поэтому не являются составной частью данной архитектуры.

В системе TSIMMIS имеются средства поддержки глобальных логических ограничений посредством менеджеров глобальных и локальных ограничений. Однако, при интеграции автономных источников информации обычно имеется мало возможности исправлять несовместимости, если ограничения нарушаются. Предполагая, что локальные ограничения

накладываются и проверяются источниками данных, любое глобальное ограничение должно определяться таким образом, чтобы оно ни каким образом не могло нарушиться.

Глобальная модель данных в системе Garlic также отличается от глобальной модели данных TSIMMIS. В то время, как описания источников данных в системе Garlic фиксированы, архитектура системы TSIMMIS является намного более гибкой.

1.3.1. Модель обмена метаданными

В работе [9] предложена само-описывающаяся глобальная модель данных, называемая Object Exchange Model (OEM, Модель обмена объектами).

Данная модель может использоваться для описания данных без необходимости в уровне описания явной схемы. Это облегчает управление доступными посредниками и адаптерами, так как не требуется, чтобы посредник поддерживал заранее описанную глобальную схему. Модель OEM основывается на структурированном графе, который формируется вложенными кортежами следующей формы < label, type, value >[23], в которой value может быть значением конечной вершины или ссылкой на вложенный кортеж. Пример данного графа приведен в таблице 1.1

Таблица 1.1 – OEM пример запроса[7]

Исходная модель (посредник Biblio):	Запрос:
< bib, set, {doc_1, doc_2, . . . , doc_n} > doc_1 : < doc, set, {au_1, top_1, cn1} > au_1 : < authors, set, {au^1_1} > au_{11} : < author-ln, str, “Ullman” > top_1 : < topic, str, “Databases” > cn_1 : < local-call#, integer, 25 > doc_2 : < doc; set, {au_2, top_2, cn_2} > au_2 : < authors, set, {au^1_2, au^2_2, au^3_2} > au^1_2 : < author-ln, str, “Aho” > au^2_2 : < author-ln, str, “Hopcroft” > au^3_2 : < author-ln; str; “Ullman” > top_2 : < topic, str, “Algorithms” > cn_2 : < dewey-decimal#, str, “BR273” >	SELECT bib.doc.topic FROM Biblio WHERE bib.doc.authors.author-ln = "Ullman"
	Результат запроса:
	< answer, set, {o_1, o_2} > o_1 : < topic, str, “Databases” > o_2 : < topic, str, “Algorithms” >

Семантика запроса основывается на метке. Моделирующая семантика в модели OEM является не более совершенной, чем в реляционной модели: само-описываемые имена используются для свойств in-line при описании данных. Нет возможности моделировать явные взаимосвязи между классами, иерархии классов, а также нет возможности добавлять дополнительные логические ограничения, как это можно делать в языках RDF-Schema and OWL. Языком глобальных запросов в TSIMMIS является язык OEM-QL. С точки зрения структуры, как OEM, так и OEM-QL[22] являются даже сходными с XML и XPath/XQuery (без использования явной XML Schema). Пример запроса на языке OEM-QL и соответствующая исходная модель приведена на рисунке. 1.4.

Как показывает данный пример, обработка запросов в TSIMMIS основывается на сопоставлении путей в графе. Для запроса, показанного на рисунке 1.4, посредник с именем Biblio должен найти все пути к структуре под-объектов, описанной последовательностью { bib, doc, authors, authors-ln }, в которой последний объект имеет значение “Ullman”. Для каждого из этих путей, подструктура bib.doc расширяется для получения всех значений для bib.doc.topic.

1.3.2 Обработка запросов в системе TSIMMIS

Аналогично системе Garlic, могут быть использованы исходные возможности обработки запросов имеющиеся у информационных систем, имеющих адаптеры. Однако в системе TSIMMIS используется другая методология, которая основывается на включении запросов. Возможности адаптера описываются параметризованными правилами [7]. В этом смысле, возможности (эти же правила) также описывают, какие кортежи предоставляют источники данных. В ходе обработки запросов, эти правила сопоставляются с полученным (обрабатываемым) запросом. В отличии от системы Garlic, эти правила не являются описанными так подробно, как описание алгебраических операций. В худшем случае адаптер (или посредник) могут поддерживать только конкретные запросы, и если некоторая часть запроса отсутствует, то адаптер не может ответить на него. Однако, если определен набор правил, а

ответ на запрос не может быть получен напрямую, то адаптер будет пытаться объединить правила для поиска эквивалентного решения . Этот процесс аналогичен вычислению включения запросов, который используется для разрешения (формирования) представлений в реляционных системах управления базами данных. Оптимизатор для системы выполнения TSIMMIS посредников был предложен в [7]. Это алгебраический оптимизатор, который является частью расширителя представлений и оптимизатор на основе стоимостей, который определяет, какие запросы отправлены конкретным источникам и в каком порядке они посланы. Он использует подход в значительной степени отличающийся от подходов используемых обычными, основанными на стоимостях, оптимизаторами для планов реляционной алгебры (например, оптимизатора Starburst).

Во многом это связано с тем, что данный подход обработки запросов основывается на правилах и в значительной степени связан с логическим программированием. Данная оптимизация является более трудной, так как модель OEM использует вложенные объекты, которые имеют не известную структуру. Например, когда посредник выполняет связывание, то невозможно предсказать, будет ли конкретное значение получено от того или другого источника, что делает проталкивание выборок вниз не возможным. Гибкая архитектура системы TSIMMIS, основанной на моделе OEM и языке запросов OEMQL явно имеет узкое место, связанное с реализацией и применением сложной оптимизации запросов. Это является основной причиной, почему данный подход является подходящим только до определенной степени.[25]

1.3.3 Основанный на Web просмотр графов ответов модели OEM

Достаточно интересно, как в 90-е годы были предложены первые подходы к использованию WWW, в качестве основы для интерфейса пользователей распределенных систем. В качестве части проекта TSIMMIS был разработан Information Explorer (Обозреватель Информации), основанный на системе Mosaic (Mosaic-based Information Explorer, MOBIE). Хотя он был очень простым, в сравнении с современными интерфейсами, но по крайней мере он

был первым подходом к навигации по графу взаимосвязанной информации на основе концепции гипермедиа. Для пояснения смысла OEM меток, MOBIE имел специальную кнопку для вызова помощи, которая могла использоваться для вызова тектовых описаний. Это очень похоже на свойство rdfs:comment, используемое для описания элементов RDF схемы на естественном языке. В сравнении с расширяемым языком RDF (Resource Description Framework), выразительность модели OEM является низкой. Запросы на языке OEM-QL начинаются с корневой вершины, что не требуется при выполнении запросов к RDF графам. Язык RDF лучше подходит для описания распределенной информации, так как он не использует предположение об уникальности имен и основывается на предположении об открытости мира.

1.4 Компонент распределенного поиска информации (DISCO)

Система DISCO (Distributed Information Search Component), разработанная в Европейском проекте [26] разрабатывалась, как компактная и простая система.

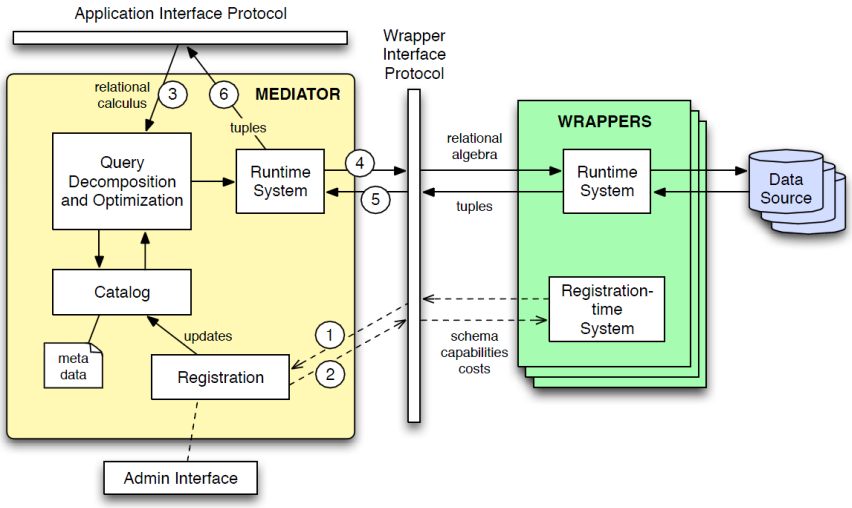


Рисунок 1.4 – Архитектура DISCO

На рисунке 1.4 числа показывают порядок выполнения работы, начиная с регистрации нового адаптера источника данных в посреднике DISCO, предоставление адаптером всей требуемой информации, такой, как экспортируемая схема, описания возможностей и стоимостных функций и процесс выполнения запросов. На данном рисунке также показано, что запросы

на языке глобальных запросов OQL обрабатываются на основе реляционного исчисления, однако, подзапросы основываются на отдельных алгебраических операциях. Например, адаптер может предоставлять функцию доступа, но не операцию связывания в описании своих возможностей. Результатами работы адаптера всегда являются кортежи.

Особое внимание было уделено проблеме не доступности источников данных. Система DISCO может предоставлять частичные результаты, когда источник данных становится не доступным в ходе обработки запросов. Данные, которые не удалось получить, сохраняются в форме некоторого запроса, который может быть выполнен позже для получения полного результата. Также, как и система Garlic, DISCO использует стандарт ODMG-93 для описания глобальной модели данных.[28] Однако. Вместо создания собственного языка запросов, глобальные запросы в DISCO описываются на языке запросов ODMG, который называется OQL (Object Query Language). В качестве специальной возможности, система DISCO показывает метаданные о зарегистрированных адаптерах, к которым могут направляться запросы с помощью OQL, как к обернутым источникам данных.

1.4.1 Поддержка отображения моделей

В проекте DISCO также решалась задача отображения моделей. Система DISCO предоставляет три метода для интеграции источников данных, имеющих локальные схемы, которые не соответствуют описанию глобальной схемы. Однако предлагаемый подход работает только между схемами ODMG. Таким образом, предполагается, что на первом этапе, адаптер уже предоставляет ODMG схему для унаследованных источников базовой информационной системы.

В системе предлагаются следующие три метода интеграции [9]:

- задание (определение) подтипов для моделирования (subtyping for modeling) аналогично подструктурам,
- отображения для моделирования сходных структур и
- представления для моделирования не однородных структур.

Метод определения подтипов уже задан в стандарте ODMG и хорошо используется для объектов разного типа, но сходной структуры. Например, если глобальная модель данных определяет только интерфейс Person, а источник данных имеет интерфейс Student со сходной подструктурой, адаптер должен определить отношение подкласса для того, чтобы включить класс Student в качестве подкласса глобального типа Person. Глобальный запрос для получения информации о людях будет включать описания всех студентов источника данных. Определение подтипов обычно используется вместе с отображением.

Отображение в системе DISCO описывается в виде части определения интерфейса. Имеются два разных вида отображений:

- отображение между локальными типами (например, отношением – relation) и глобальным типом,
- отображение между полем локального типа и полем глобального типа.

Например, если поле определяющее имя студента в типе Student отличается от поля содержащего имя в типе Person, то может использоваться отображение 1:1 (один к одному) для связывания полей с именами. В системе DISCO используются только плоские отображения. Нет возможности выполнять связывание между разными вложенными структурами и нет способа разрешать конфликты схемной разнородности. Также было только объявлено о поддержке функциональных отображений, но очевидно не было реализовано.

Однако, во многих случаях разнородных структур типов, понятие представлений является очень мощным подходом для согласования типов. В действительности представления являются выражениями, описывающими запросы на языке OQL, которые могут генерировать необходимый виртуальный интерфейс для согласования с определением глобальных типов. Более того, представление может ссылаться на другие типы. Если не учитывать вопросы производительности, то это предоставляет мощный подход, когда не достаточно более простых отображений.

1.4.2 Обработка запросов и стоимостные функции

Разработчики системы DISCO могли получить выгоду от ранее приобретенного опыта в ходе выполнения проекта IRO-DB при создании средств обработки запросов и оптимизации. Архитектура проекта IRO-DB требовала, чтобы каждый адаптер поддерживал полный язык OQL для ответа на локальные запросы. В результате этого, для каждого адаптера требовалось реализовать процессор запросов на языке OQL (OQL query processor), работающего над исходными источниками информации. [26]

Для локальных запросов отправляемых адаптерам, система DISCO использует алгебраические операторы аналогичные тем, что использовались в системе Garlic, вместо полного языка декларативных запросов. Такой подход позволяет реализовать более детальную модель стоимости и улучшить оптимизацию запросов. Оптимизатор системы DISCO основывается на стандартных методах оптимизации реляционной базы данных. Все планы имеют глобальную часть и множественные локальные подпланы, которые непосредственно отправляются соответствующим адаптерам. Для элемента PushDown системы Garlic есть эквивалентный оператор в DISCO, который называется submit. Много усилий исследователей в проекте DISCO было посвящено улучшению стоимостной модели [10].

1.5 Система InfoSleuth

Проект InfoSleuth (информационная ищейка) был инициирован когда-то существующей корпорацией «Microelectronics and Computer Technology Corporation (MCC)» в 1995, как продолжение проекта Carnot [4]. Проект Carnot был очень ранним подходом (и возможно самым первым) к созданию крупно-масштабной, много-агентной программной системы интеграции информации предприятий. В то время, предложенные и реализованные в данном проекте идеи и концепции могли рассматриваться как революционные. Как, проект Carnot, так и проект InfoSleuth используют онтологии для поддержки семантической интеграции разнородных источников данных. В проекте Carnot, используется высоко-уровневая онтология Cyc [11] для отображения сущностей

реального мира с так называемыми артикуляционными аксиомами на формально-определенные понятия онтологии Cyc. В проекте Carot было разработано несколько прототипов приложений для таких прикладных областей, как управление рабочими потоками, доступ к разнородным базам данных, поиск знаний в больших базах данных и интегрированный доступ к хранилищам текстов и структурированным базам данных.

Проект InfoSleuth спонсировался такими большими компаниями, как Andersen Consulting, Bellcore, Boeing, NCR/AT&T и т.д., а также министерством обороны США, которые были заинтересованы в безотказном (fail-safe), масштабируемом решении для интеграции информации поддерживаемой в большом разнообразии географически распределенных источников. [25]

Идея автономных программных агентов которые действуют от имени пользователей для решения некоторой особой задачи, пришла из сообщества исследователей в области искусственного интеллекта и стала достаточно популярной в 90-е годы. Программные агенты являются в значительной степени автономными компьютерными программами (software program), которые общаются по сети (например, сети Интернет) с другими программными агентами.

На рисунке 1.6 показана обобщённая схема архитектуры InfoSleuth. В мультиагентной среде, такой, как InfoSleuth, в основном имеются два вида агентов: пользовательские агенты и другие, полностью независимые агенты с очень специфическими возможностями.

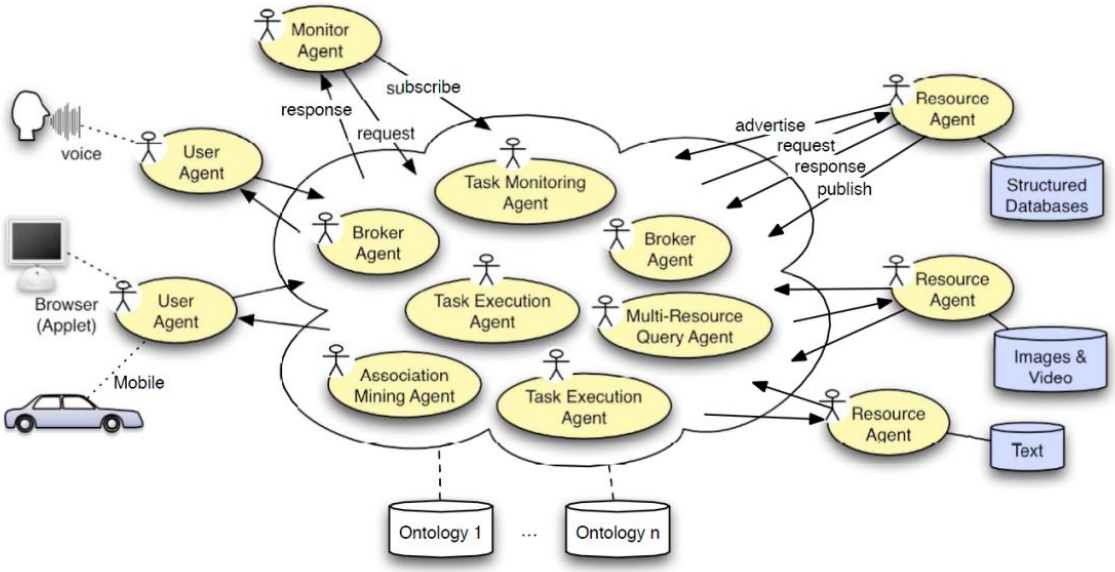


Рисунок 1.5 – Обобщенная схема системы InfoSleuth, включающая агентов, релевантных для интеграции информации.

Например, есть такие агенты, как:

- отслеживающие агенты,
- связывающие агенты,
- выполняющие задачи агенты,
- отслеживающие выполнение задач агенты,
- анализирующие взаимосвязи агенты,
- агенты, выполняющие запросы к набору ресурсов,
- агенты ресурсов (поддерживающие ресурс агенты)
- и другие.

Пользовательский агент используется в качестве интеллектуального интерфейса для людей – пользователей. Данная система была реализована на языке Java для обеспечения независимости от платформы. Пользовательский агент также может быть интегрирован в Web-сайты в виде Java апплетов. Также, как все другие агенты, он оповещает о своем существовании и возможностях связывающему агенту, который действует как супер- в данной и передает сообщения другим агентам. Размещение пользовательского агента совершенно не зависит от того, где находится сам пользователь, что делает возможным использовать данную систему с любого компьютера или устройства подключенного к сети Интернет. В то время, когда разрабатывалась система InfoSleuth, еще ничего не было известно о таком пиринговом (peer-to-

peer) программном обеспечении промежуточного уровня, как например JXTA, которое было разработано фирмой Sun в 2001 [12]. Таким образом, система InfoSleuth также может рассматриваться в качестве пионерского проекта для пиринговых сетей, которые стали очень популярными в начале 20 тысячилетия (и особенно после судебного процесса над Napster, организованного RIAA (Recording Industry Association of America)).

Агенты ресурсов (resource agent, т.е. адаптеры) могут рассматриваться в качестве привратников информационных ресурсов (gatekeeper to an information source). Их задачей является реагирование (respond) на запросы, ответы на которые могут быть выведены из прикрепленных к ним источников информации. Такими источниками могут быть некоторого вида базы данных, хранилища мультимедия или просто коллекции текстовых документов. Все источники информации в системе InfoSleuth являются полностью автономными. Внешне они представляются на уровне релевантных им семантических понятий и в связи с этим их локальная схема, формат и представление являются полностью независимыми.

Агент ресурса является эквивалентом адаптера в архитектуре посредников-адаптеров. Он также поддерживает отображение общих онтологий предметных областей на локальную схему и язык запросов, являющийся «родным» для базового ресурса. Агент ресурса оповещает о своих возможностях: т.е., что он знает и какого вида ответы он может формировать, и какие запросы он может принимать для данного ресурса.

В архитектуре данной системы не просто найти эквивалент для посредника, так как подкомпоненты посредников, такие, которые есть в системах Garlic, TSIMMIS или DISCO, разделены и выполняются разными агентами. Посредник, как он понимается в системе Garlic может состоять из связывающего агента, агент выполнения задач, агента онтологий, агента выполнения запросов к набору ресурсов и агента отслеживания выполнения задач.[23]

1.6 Система интеграции SemWIQ

Еще одним исследованием в направлении интеграции больших объемов данных является система SemWIQ, основанная на технологиях Семантической Паутины.

Основными составными элементами являются клиенты, показанные над посредником, который находится в центре, и несколько источников данных, которые показаны в нижней части рисунка 1.6.

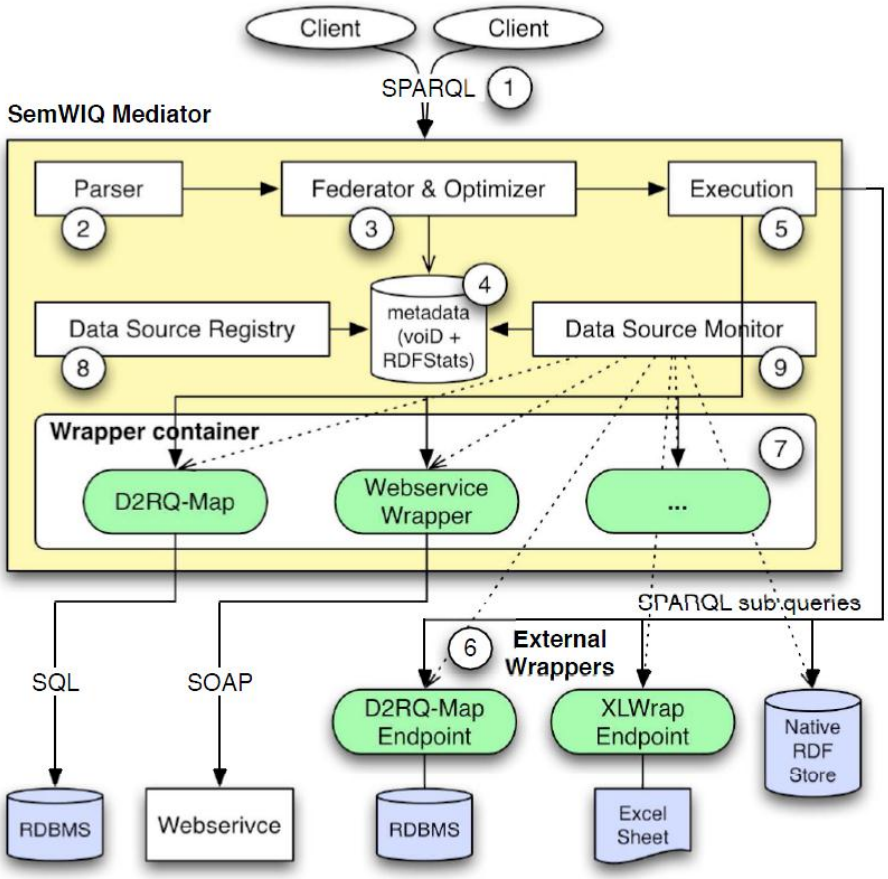


Рисунок 1.6 – Архитектура посредников-адаптеров в подходе SemWIQ [13]

Пользователи системы SemWIQ выполняют запросы к посреднику с помощью клиента (самостоятельная программа или web-приложение) передают на обработку глобальные SPARQL запросы (1). Глобальный SPARQL запрос может содержать не только любые терминологические понятия, такие, как классы и свойства, но также и экземпляры понятий, которые существуют в глобальном виртуальном графе. [13]

Грамматический разборщик запросов (2) формирует канонический план запроса, который преобразуется и оптимизируется интегратором (3) на основе информации, которая содержится в каталоге метаданных (4).

В системе SemWIQ доступны два разных федератора. Разделенный на части и оптимизированный план запроса окончательно обрабатывается подсистемой выполнения запросов. Глобальный план запроса состоит из нескольких локальных подпланов, которые далее объединяются высокоуровневыми алгебраическими операциями, которые выполняются в посреднике. Любая операция, которая может быть непосредственно выполнена адаптером удаленно включается в план локального запроса. [29]

Удаленные адаптеры размещаются непосредственно над удаленными источниками данных (6). Для источников данных, которые являются полностью автономными, соответствующие адаптеры могут помещаться в специальный контейнер, который находится в посреднике (7).

Журнал источников данных (8) за регистрацию и де-регистрацию источников данных. Источник данных может быть зарегистрирован и де-регистрирован путем отправки HTTP POST запроса с соответствующим URI конечной точки SPARQL. Монитор источников данных (9) периодически выполняет проверку доступности зарегистрированных источников данных собирает метаданные в формате voiD и текущие статистические данные RDFStats.

2. Объект и методы исследования

Использование Semantic Web или онтологий регулярно рассматривается и в научной среде, и в среде практиков. Существует достаточно много теоретических и практических работ, прямо или косвенно, затрагивающих использование онтологий для упорядочивания и стандартизации информации, исходя из требований разработки RDF-графов, принятую в Организацией W3C:

- наличие простой модели данных

- наличие формальной семантики и доказуемого логического вывода
- использование расширяемых словарей на основе URI
- использование синтаксиса на основе XML
- поддержка использования типов данных XML схемы
- возможность любому делать объявления о любом ресурсе[15]

2.1. Подход Semantic Web

На сегодняшний день Интернет стал персональным. Интернет все больше знает о его пользователях. Отчасти, пользователи сами дают знания, публикуя свои личные данные в социальных сетях, пользуясь распространенными поисковыми системами, пройдя авторизацию. Из этого следует, что уже завтра, вводя в строку поиска «Где помыть автомобиль недорого», пользователь будет получать ответ в виде ближайшей автомойки и её местоположение в результате четкого ответа на четкий вопрос. Теперь не нужно будет переходить по большому количеству ссылок из выдачи поисковой системы различных поисковиков, разочаровываясь в том, что очередная открытая вкладка – это очередной дорогой салон, продвигаемый силами SEO специалистов. [27]

Это касается различных сфер жизни и деятельности человека – начиная от бытовых и заканчивая более глобальными. Например, покупка автомобиля или квартиры, поиск работы и другие.

Более того, поисковая система сможет определить, какой именно автомобиль нужен пользователю на основе информации о том, какими тест-драйвами он больше всего интересуется и какие автомобильные сайты посещает, в каком районе и в каком ценовом диапазоне вы хотите найти квартиру, не голодны ли вы, какую еду предпочитаете и так далее.

С развитием семантического веба после сбора определенных данных о пользователе технологии позволят составить его социально-демографический

портрет. Собранные пользовательские данные компьютеры будут понимать уже как портрет личности.

Во многом такой динамике способствует стремление упростить сервисы и сделать упрощенный доступ пользователей к контенту. Ставшая модной в последняя время, авторизация через социальные сети (Вконтакте, Facebook), специальные сервисы (OpenID, OAuth), комментирование через виджеты социальных сетей[13].

2.2. Resourse Description Framework

RDF — это универсальный метод разделения знания на маленькие части, в соответствии с некоторыми правилами, учитывающими семантику (смысл) этих частей. Суть в том, что такой метод должен быть достаточно простым, чтобы с его помощью можно было описать любой факт, и достаточно структурированным, чтобы представить факт в такой форме, в которой компьютерные приложения смогут осуществлять полезные действия со знаниями, выраженными в формате RDF[14].

Стандарт RDF описывает, как с помощью объединения трёх таких сущностей можно констатировать некоторый факт. Значение триплета '(Джон, Боб, отношение дружбы)' заключается в том, что Джон и Боб являются друзьями. Совокупность фактов в достаточном количестве может являться некоторой формой представления знаний. Стандарты, основанные на RDF, включая RDFS и OWL, дополняют семантику RDF, делая возможным построение логических выводов на основе совокупности данных.

RDF обеспечивает универсальный, гибкий метод декомпозиции знаний на маленькие части, называемые триплетами, по некоторым правилам, учитывающим семантику (смысл) этих частей. Основа метода в разбиении знаний на части и составлении того, что принято называть помеченным направленным графом. Каждая дуга такого графа представляет собой некоторый факт, отношение между двумя сущностями.

Факт (или утверждение), представленный таким способом, состоит из трёх частей: субъект, предикат (аналогичен глаголу в естественных языках) и

объект. Субъект представляется узлом, из которого исходит дуга. Значение предиката определяется типом дуги (обозначается с помощью метки дуги). Объект – это узел, к которому направлена дуга.

Данные RDF представляются в виде RDF Schema(RDFS). RDFS представляет собой словарь для описывания RDF триплетов. Использование RDFS делает возможным обрабатывать данные в RDF графе.

Многие части словаря RDFS вошли в OWL. OWL тоже представляет собой словарь. Эго основной функцией является описывание ресурсов, относящихся к веб-страницам.

2.3. Язык запросов SPARQL

SPARQL является языком запросов к RDF документом. Он принят как стандарт Консорциумом Всемирной паутины. Создание точек доступа SPARQL рекомендуется при создании и публикации веб-страниц

2.4. Реляционные базы данных

Реляционная база данных — это совокупность взаимосвязанных таблиц, каждая из которых содержит информацию об объектах определенного типах[15]. Строка таблицы содержит данные об одном объекте (например, товаре, клиенте), а столбцы таблицы описывают различные характеристики этих объектов — атрибутов (например, наименование, код товара, сведения о клиенте). Записи, т.е. строки таблицы, имеют одинаковую структуру — они состоят из полей, хранящих атрибуты объекта. Каждое поле, т. е. столбец, описывает только одну характеристику объекта и имеет строго определенный тип данных. Все записи имеют одни и те же поля, только в них отображаются различные информационные свойства объекта. [16]

2.4. Предлагаемое подход к решению исследуемой задачи

Предложено использовать следующие методы и модели для интеграции реляционных баз данных:

- Преобразование реляционных баз данных в RDF
- Создание алгоритма интеграции онтологий из RDF графов, образованных баз данных

- Разработка и реализация серверной части веб-сервиса
- Разработка и реализация клиентской части веб-сервиса

3. Разработанное решение

3.1. Анализ решения

Для реализации механизма интеграции реляционных баз данных необходимо детально разработать архитектуру приложения или сервиса. В первую очередь необходимо определить функции, которые будет выполнять программный продукт, выявить необходимые требования.

Разработка архитектуры программного обеспечения это вид деятельности на этапе проектирования ПО, связанный с определением основных ограничений, накладываемых на проектирование системы, такие как выбор парадигмы программирования, архитектурных стилей, стандарты разработки ПО. Они основаны на использовании компонентов, принципов проектирования и ограничения, накладываемые государственным законодательством. Детальное проектирование, то есть разработка тактики — это деятельность, связанная с определением локальных ограничений проекта, такие как шаблоны проектирования, архитектурные модели, идиомы программирования и рефакторинга[17].

3.2. Архитектурный стиль приложения

Основываясь на функциональных требованиях и упрощенной модели системы можно сделать заключение: информационная система должна быть выполнена в виде веб-сервиса.

Это объясняется тем, что веб-приложения на данный момент являются самыми кросс-браузерными приложениями. Для их использования необходим лишь веб-браузер. На рисунке 3.1 можно увидеть самые распространенные операционные системы для персональных компьютеров и мобильных устройств, а также самые распространенные браузеры. [21]

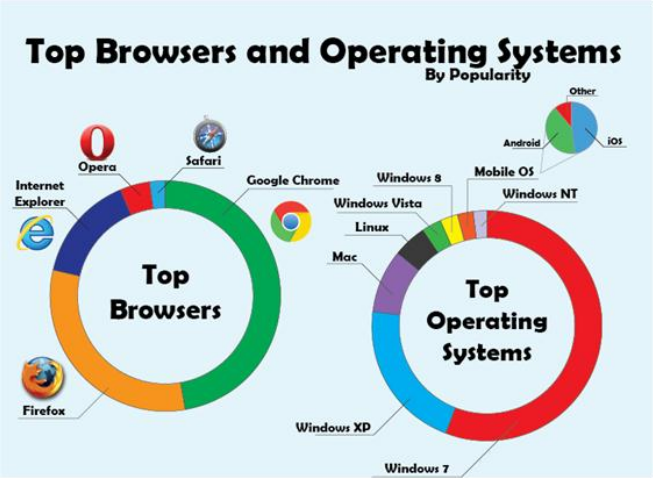


Рисунок 3.1.-.Диаграмма использования ОС и браузеров

Самые популярные браузеры, а именно FireFox и Google Chrome, имеют реализацию на самых, используемых платформах: семейства Windows, Linux и MacOS, Android и iOS. Следовательно, реализация поставленной задачи по интеграции в виде веб-приложения учитывает сразу несколько функциональных требований: доступность широкому кругу пользователей, отсутствие необходимости устанавливать дополнительное программное обеспечение.

Учитывая основную функцию информационной системы можно сделать вывод, что лучшим архитектурным стилем для её реализации является сервисо-ориентированная архитектура. Она предусматривает наличие модулей, которые слабо связаны между собой и легко заменяемы. Взаимодействие реализуется с помощью различных интерфейсов и протоколов

3.3. Описание модели системы

Следующим этапом проектирование является создание модели системы. Этот этап необходим для определения необходимого программного обеспечения: языка программирования, шаблонов проектирования, программных библиотек, среды разработки и платформы для будущей публикации. Простая модель информационной системы должна затрагивать лишь основные пункты её работы. Также необходимо учесть информационные потоки, которые передают необходимые данные между различными

участниками информационной системы. Для начала стоит выделить основные пункты работы системы:

1. Получение доступа к реляционным базам данных
2. Представление реляционных баз данных в виде RDF графов
3. Анализ полученных RDF -графов
4. Интеграция данных
5. Создание RDF графа на основе интеграции
6. Вывод результата

На рисунке 3.2 представлена пример модели информационной системы. С помощью клиентской части приложения пользователь может инициировать запрос к серверу. Сервер запрашивает данные с реляционных баз данных. Данные возвращаются через посредника. Посредник преобразует реляционные базы данных в RDF графы. На сервере происходит интеграция полученной информации. И пользователю получает ответ.

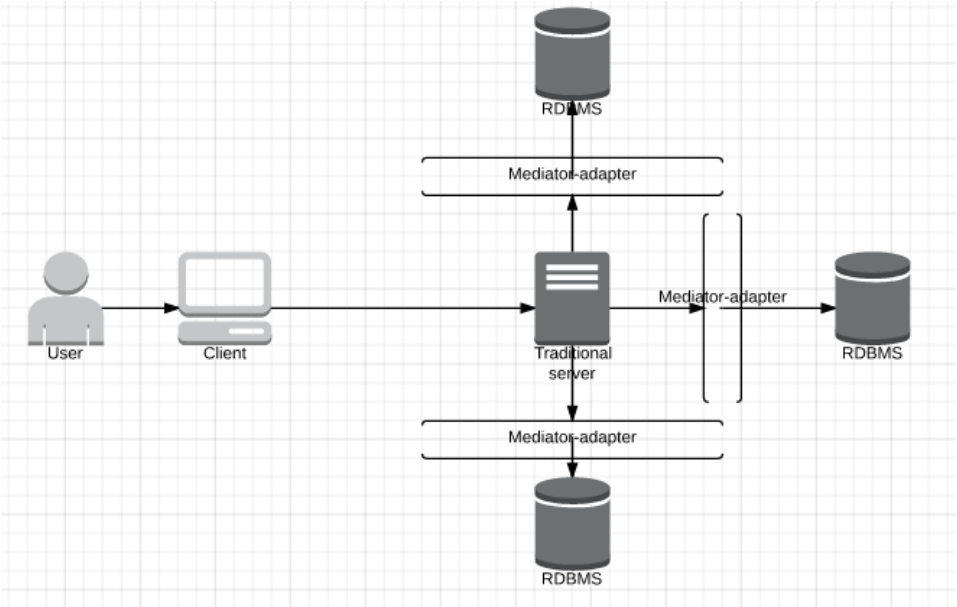


Рисунок 3.2 – Пример модель интеграции реляционных БД информационной системы

3.4. Выбор программных средств

Для реализации информационной системы в виде веб-приложения необходимо выбрать программные средства, которые реализуют интеграцию

реляционных баз данных, используя Semantic Web, с возможностью развертывания приложения в сети Интернет.

3.4.1. Посредник-адаптер

Одной из основных задач является преобразование реляционной базы данных в RDF – граф. Она будет решена с помощью посредника-адаптера. Таким медиатором может выступать платформа D2RQ

Платформа D2RQ

Для создания адаптеров-посредников в разрабатываемой информационной системе, как и в подходе SemWIQ, можно использовать платформу D2RQ

Платформа D2RQ — это информационная система для предоставления доступа к реляционным база данным как виртуальные RDF графы, доступные только для чтения. В частности, доступ к RDF содержимому реляционных баз данных происходит без их преобразования в RDF хранилища.

Используя платформы D2RQ возможно:

- совершать запросы к реляционной базе данных, используя язык запросов SPARQL
- получать доступ к реляционным базам данных через Web-сеть как набор Связанных данных
- создавать свои RDF дампы для последующей загрузки их в RDF хранилища
- работать с информацией из реляционных баз данных, используя различные программные библиотек для работы с RDF.

Состав D2RQ-платформы:

- generate-mapping
- d2r-server
- d2r-query
- dump-rdf
- D2RQ+Jena API.

На рисунке 3.3 показана архитектура платформы D2RQ

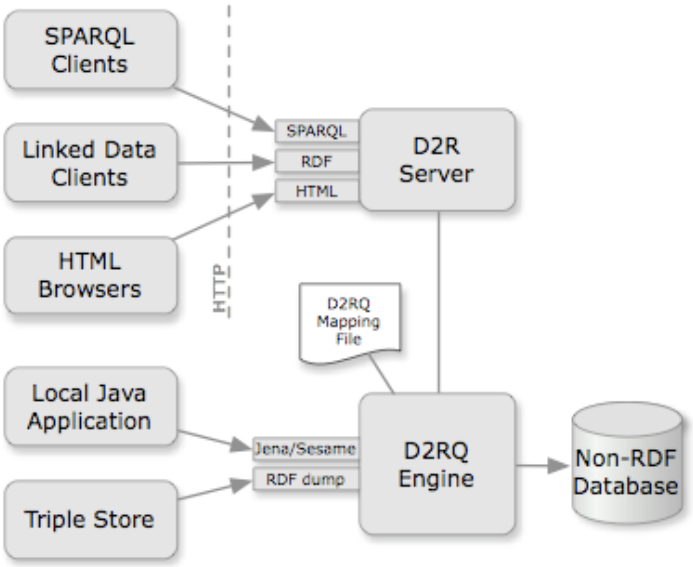


Рисунок 3.3 – Архитектура платформы D2RQ

Формирование отображения

Инструмент generate-mapping (формирование отображения) создает файл сопоставления D2RQ путем анализа схемы существующей базы данных. Этот файл описания отображения, называется default mapping, он отображает каждую таблицу как RDFS класс, который основан на имени таблицы и отображает каждый столбец в собственности, основанной на имени столбца. Этот mapping файл может быть использован как есть или может быть изменен.

Сервер D2R

D2R-сервер является инструментом для публикации реляционных баз данных на технологии Semantic Web. Это позволяет RDF и HTML-браузерам осуществлять навигацию содержания базы данных, а также позволяет запрашивать базу данных с помощью языка запросов SPARQL.

Данные о Semantic Web моделируются и представляются в RDF. D2R сервер использует файл отображения (mapping файл) для представления D2RQ содержимого базы данных, и дает доступ к данным RDF для просмотра и поиска - две основные парадигмы доступа к Semantic Web.

SQL запросы из Web переформируются с помощью generate-mapping. Это позволяет публиковать RDF от крупных рабочих баз данных и убирает необходимость обработки данных в выделенном RDF триплете.

Возможности D2R-server**:**

- Удобный просмотр содержимого базы данных

Простой веб-интерфейс позволяет производить поиск в содержимом в базах данных и предоставляет пользователям RDF - данных в удобном формате при предварительном просмотре.

- Составление удобных URI (Uniform Resource Identifier)

На основе принципов Linked Data, D2R сервер назначает URI для каждого объекта, описанного в базе данных,то есть, описание RDF можно восстановить через владельца URI. Semantic Web браузеры, например, Marbles или LinkSailor могут использовать URI для навигации по Semantic Web.

- Высокая согласованность

D2R-server создает единообразные URI, что облегчает работу с ними, используя другие платформы. Также в D2R-sever реализованы общепринятые интерфейсы, например HTTP.

- Поддержка SPARQL

Позволяет запрашивать данные через SPARQL запросы через язык запросов - SPARQL 1.1. Также имеет простейший встроенный текстовый редактор SPARQL запросов.

- Работа с URI

D2R-server позволяет не только создавать URI, но также и изменять существующие, конфигурировать файлы с хранящимися в них триплетами.

Компонент D2R-query

Компонент D2R-query позволяет выполнять запросы SPARQL через D2RQ-карту реляционной базы данных из командной строки. Это может быть сделано с использованием или без mapping файла. Если mapping файл задан, то инструмент будет запрашивать виртуальный RDF граф, определенный mapping

файлом. Если файл отображения не указан, то инструмент будет использовать отображение по умолчанию.

Компонент Dump-RDF

Компонент Dump-RDF создает D2RQ файл с содержимым всей базы данных в одном файле RDF. Это может быть сделано с или без файла отображения D2RQ. Если файл отображения задан, то инструмент будет использовать его, чтобы перевести информацию баз данных RDF. Если файл отображение не указан, то инструмент будет использовать отображение по умолчанию.

Точка доступа D2QR + Jena API

Jena — это Java фреймворк для создания приложений, работающих с Semantic Web. D2RQ может использоваться как компонент для получения доступа к RDF.

D2RQ позволяет Jena извлекать данные на разных уровнях:

1. Jena Model API
2. Jena Graph API
3. SPARQL запросы к RDF.

3.4.2. Серверная и клиентская части

Для решения задачи по реализации алгоритма интеграции и серверной части приложения необходимо использовать мощный инструмент, который предусматривает возможность работы с RDF графами, реализует много поточность, а также обеспечит функционирование веб-сервиса. Таким инструментом является язык программирования Python. [18]

Язык программирования Python

Python — язык программирования высокого уровня. Он предназначен для выполнения задач общего назначения. Python ориентирован на высокую скорость написания кода, а также удобность его чтения. Стандартная библиотека Python содержит большой и разнообразный набор методов, хотя ядро Python исполняет только основные функции, которые характерны для языков программирования высокого уровня.

Python — мультипарадигмный язык программирования. Он поддерживает:

- структурное программирование
- объектно-ориентированое программирование
- функциональное программированием
- императивное программированием
- аспектное-ориентированное программирование

Также Python имеет динамическую типизацию, автоматическое управление памятью, обработку ошибок, полную интроспекцию и возможность создания многопоточных программ. Его широта использования и простота изучения получили широкий отклик не только в профессиональной среде, но и в научной. Благодаря этому количество библиотек и фреймворков для Python постоянно пополняется. Среди них можно выделить библиотеки, которые нашли применение в научной среде:

1. SciPy — библиотека содержащая пакеты для интегрирования, генетических алгоритмов, решения дифференциальных уравнений и других задач, решаемых в науке или при инженерной разработке
2. Matplotlob — библиотека для визуализации 2D и 3D графики
3. NumPy — расширение, которое добавляет поддержку многомерных массивов и матриц, а также многие математические функции необходимые для работы ними

На основании широкого применения Python в научной среде, а также наличия в нем большого количества необходимых программных средств в виде библиотек и фреймворков, Python ляжет в основу проектируемой программной системы.

Фреймворк Django

Для создания веб-приложения будет использоваться Django Python Framework. Планирование и реализация Web-сайтов всегда сопровождается большими затратами усилий. Django представляет собой один из лучших на сегодняшний день фреймворков, который позволяет быстро вести разработку

высокопроизводительных и полнофункциональных сайтов. С помощью django легко выстаиваются масштабируемые и легко расширяемые приложения для Web с дизайном любой степени сложности.

Абстрагируясь от низкоуровневого процесса Web-разработки, django позволяет разработчикам быстро создавать основанные на базах данных динамичные Web-сайты. Одним из основных преимуществ django является переносимость созданных на ее основе продуктов в силу переносимости их базиса – языка высокого уровня Python.

Django включает в себя Model View Controller (MVC) – шаблон проектирования, позволяющий разделить общую архитектуру на отдельные части. При этом управляющая логика разделена на три отдельных компонента так, что модификация одного из них оказывает минимальное воздействие на другие части. К таким компонентам относят разделяемые данные, логику и слои визуализации. В общем случае такая концепция позволяет разделить разработку информационного наполнения на уровне базы данных и разработку Web-страниц.

Django базируется на классе Python django.db.models.Model, который задает данные модели так, чтобы они были пригодны к использованию на Web-сайтах. Эти данные определяются соответствующими атрибутами объектов, которые сохраняются в базе данных в процессе работы. При создании сайта создается подкласс класса Model и добавляется поле членов в класс для задания специфических данных.

Интерфейс модели django обеспечивает многоплановый выбор из всех имеющихся типов моделей для отбора наиболее подходящего в данной ситуации. Выбранная модель проекта синхронизируется с работающей базой данных, где она сохраняет свои данные в таблицах. Django вообще обеспечивает хороший интерфейс к базам данным, позволяющий получать надежный доступ к информации из слоев визуализации и шаблонов.

Изменение отображения содержимого в зависимости от принимаемого URL-запроса является многоступенчатым процессом. Когда django-сервер

получает URL-запрос, он парсит его и, используя предыдущие установки шаблонов, определяет, какой участок кода Python будет выполняться для требуемого отображения.

Парсер шаблонов в django позволяет самостоятельно настраивать свои шаблоны, которые используют функции отображения Web-страниц при построении ответа на URL-запросы. Это позволяет разработчикам Python сфокусироваться на создании данных, которые будут отображаться, а программистам HTML – сфокусироваться на дизайне Web-страниц.

Возможности фреймворка:

- ORM, API доступа к БД с поддержкой транзакций
- встроенный интерфейс администратора, с уже имеющимися переводами на многие языки
- диспетчер URL на основе регулярных выражений
- расширяемая система шаблонов с тегами и наследованием
- система кеширования
- интернационализация
- подключаемая архитектура приложений, которые можно устанавливать на любые Django-сайты
- «generic views» — шаблоны функций контроллеров
- авторизация и аутентификация, подключение внешних модулей аутентификации: LDAP, OpenID и проч.
- система фильтров («middleware») для построения дополнительных обработчиков запросов, как например включённые в дистрибутив фильтры для кеширования, сжатия, нормализации URL и поддержки анонимных сессий
- библиотека для работы с формами (наследование, построение форм по существующей модели БД)
- встроенная автоматическая документация по тегам шаблонов и моделям данных, доступная через административное приложение [19]

Также Django может реализовать или взаимодействовать с HTML формами и JavaScript. Эти технологии являются стандартом для создани клиентских частей веб-приложений.

Программная библиотека RDFLib

Так как в реализации приложения по интеграции реляционных данных язык программирования Java использоваться не будет, необходимо найти программную библиотеку для извлечения RDF графов во время исполнения программы. Для языка Python существует достаточно много библиотек для работы с RDF. Для внедрения в проект была выбрана библиотека RDFLib, так как:

- библиотека содержит необходимые парсеры и сериализаторы для RDF и метаданных
- библиотека является интерфейсом для работы с RDF-графами
- библиотека содержит встроенное хранилище RDF-графов
- библиотека поддерживает язык запросов SPARQL
- она является свободно распространяемой и хранится на ресурсе GitHub

Платформа для развертывания

Для публикации, развертывания и использования веб-приложения необходимо развернуть проект на любом из доступных источников, таких как облачные сервисы (Heroku, Google App Engine) или отдельном сервере. Так как языком программирования является Python в связке с Django Framework, то стоит выбрать сервис удовлетворяющим требованиям к разработке всей системы интеграции. По функциональности результаты в использовании сторонних сервисов или настройка собственного сервера ничем не отличаются. Но с экономической точки зрения разница между ежемесячной оплатой доменного имени и внесения ежемесячных платежей на сторонние ресурсы является принципиальной. Наиболее подходящим вариантов является использование Ubuntu Server 14.04 LTS в связке с пакетам nginx для быстрой

отказоустойчивой работы и uwsgi для развёртывания приложения написанного на Django

Ubuntu Server — это свободно распространяемая операционная система, основанная на Debian. Ubuntu Server поставляется с набором программ необходимых для работы сервера или рабочей станции. Также в ней есть расширенная локализация и поддержка основных процессорных архитектур.

3.5. Требования к программной реализации

3.5.1. Назначение программы

Разрабатываемое веб-приложение служит для интеграции реляционных баз-данных, используя подход Semantic Web

3.5.2. Область применения

Данный программный продукт является сервисом, позволяющим преобразовать реляционные базы данных, в которых могут быть синонимичные поля и на основании этого получить RDF граф содержащий множество триплетов, характеризующих интегрированные реляционные базы данных как одно целое.

Для проектирования архитектуры приложения необходимо определить список необходимых требований к программному продукту.

3.5.3. Функциональные требования

Функциональными требованиями является набор утверждений относительно поведения информационной системы, ограничений.

Основываясь на необходимости создания дешевой в исполнении и эксплуатации системы, а также её доступности для конечного пользователя, можно выделить следующие функциональные требования:

- Информационная система должна реализовывать интеграцию реляционных баз данных, используя Semantic Web.
- Информационная система должна быть максимально доступна широкому кругу пользователей
- Для эксплуатации информационной системы должен быть

необходим лишь базовый набор программного обеспечения, который поставляется разработчиками операционных систем

- Информационная система должна быть автономна, то есть изменение, использующихся в ней сторонних приложений не должно сказываться на работоспособности системы

Информационная система должна интегрировать удаленные источники различных реляционных баз данных

3.5.4. Требования к надежности

Работа системы должна гарантировать средний аптайм – 99 % в год. Для этого система должна быть развернута на облачном сервисе, предоставляющем платформу как услугу, либо на сервере организации-клиента под управлением последней стабильной версии Windows Server или серверной версии операционной системы Linux.

При вводе некорректной информации программная система должна выдать предупреждающее сообщение о некорректности введенных данных и предложить повторить ввод.

Восстановление системы после критических ошибок, приводящих к неработоспособности системы должно происходить в течение суток после момента возникновения ошибки.

3.5.5. Требования к эргономике и технической эстетике

Взаимодействие пользователя и программы осуществляется посредством графического интерфейса клиентской программной системы – браузера. Программа должна иметь элементы навигации. Программа должна быть выполнена в едином стиле, иметь одинаковое расположение основных элементов навигации. Программа должна поддерживать горячие клавиши подтверждения ввода – Enter и закрытия подсказок и всплывающих окон – Escape.

При вводе некорректных данных, должны выдаваться сообщения на основном языке программной системы о неправильном вводе данных и предложение ввести данные заново.

3.6. Проект системы

3.6.1. Диаграмма проектных классов

Проектные классы – это классы, описание которых настолько полно, что они могут быть реализованы [умл2 с 372]. Описание формируется путем уточнения классов анализа, которое включает в себя добавлений деталей реализации. На диаграмме 3.4 представлена диаграмма проектных классов, реализуемых в системе.

На рисунке 3.4 также показано, что информационная система состоит из двух пакетов: WebInterface и Seweb.

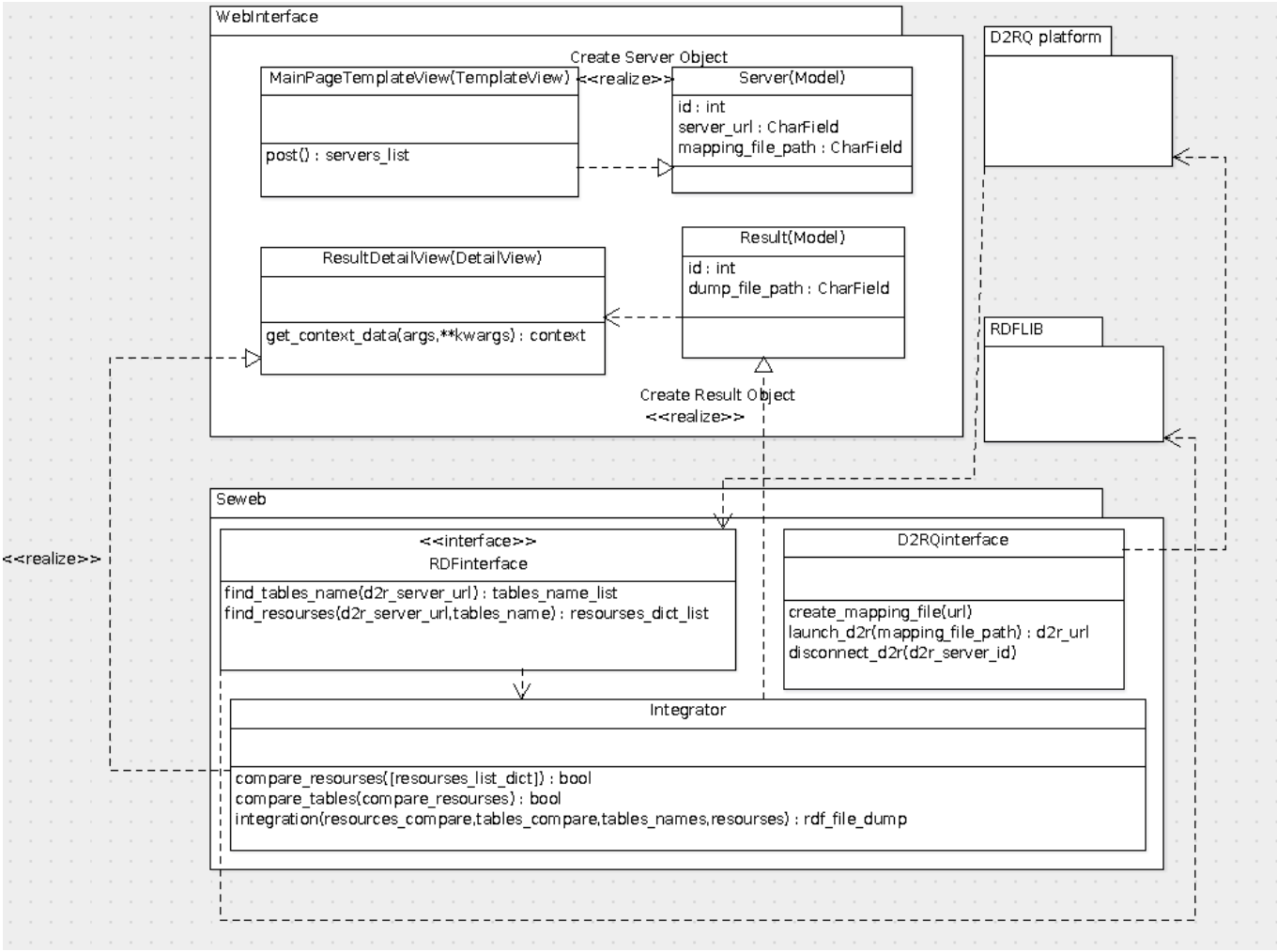


Рисунок 3.4 – Диаграмма классов системы интеграции Seweb

Пакет WebInterface содержит классы MainPageTemplateView и ResultDetailView, которые наследуются от Class Based View классов Django и

служат для обработки содержимого страницы или обработки запросов из HTML форм, присылаемых в HTTP запросах.

Пакет Seweb предназначен для работы с RDF графами: получением, сравнением и созданию новых. Он состоит из классов: D2RQinterface, RDFinterface и Integrator.

Класс D2RQinterface организует работу и разворачивание локальных D2R серверов в отдельных потоках, используя стандартную библиотеку Python для работы с потоками — GIL.

Класс RDFinterface организует содержит методы для получения RDF графов, находящихся на D2R-серверах, и данных из них с помощью RDFLib

Класс Integrator содержит методы реализующие проверки на совпадения предикатов и объектов одинакового субъекта RDF графов с разных D2R-серверов. На основании этого он может создать новый RDF граф и записать его в базу данных проектируемого приложения, или же сообщить системе, что совпадений нет.

Также стоит отметить наличие библиотеки RDF и D2RQ-платформы в информационной системе. Их наличие объясняет взаимодействие классов между собой.

3.6.2. Диаграмма вариантов использования

Диаграмма вариантов (рисунок 3.5) использования показывает возможности пользователя в системе и основывается на функциональных требованиях к разрабатываемой системе.

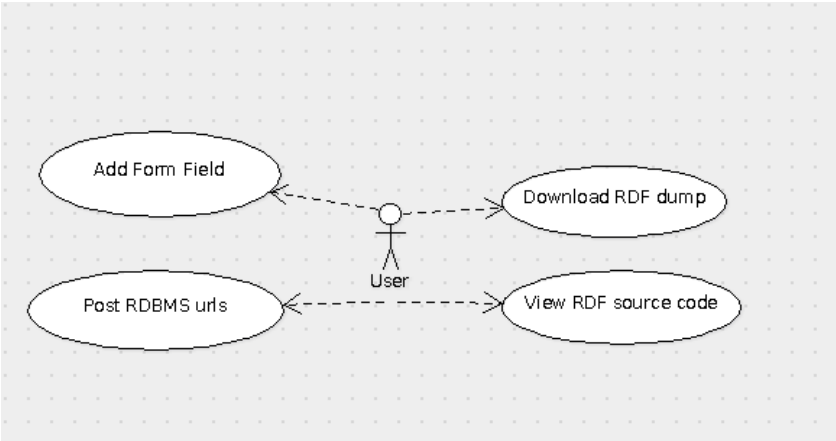


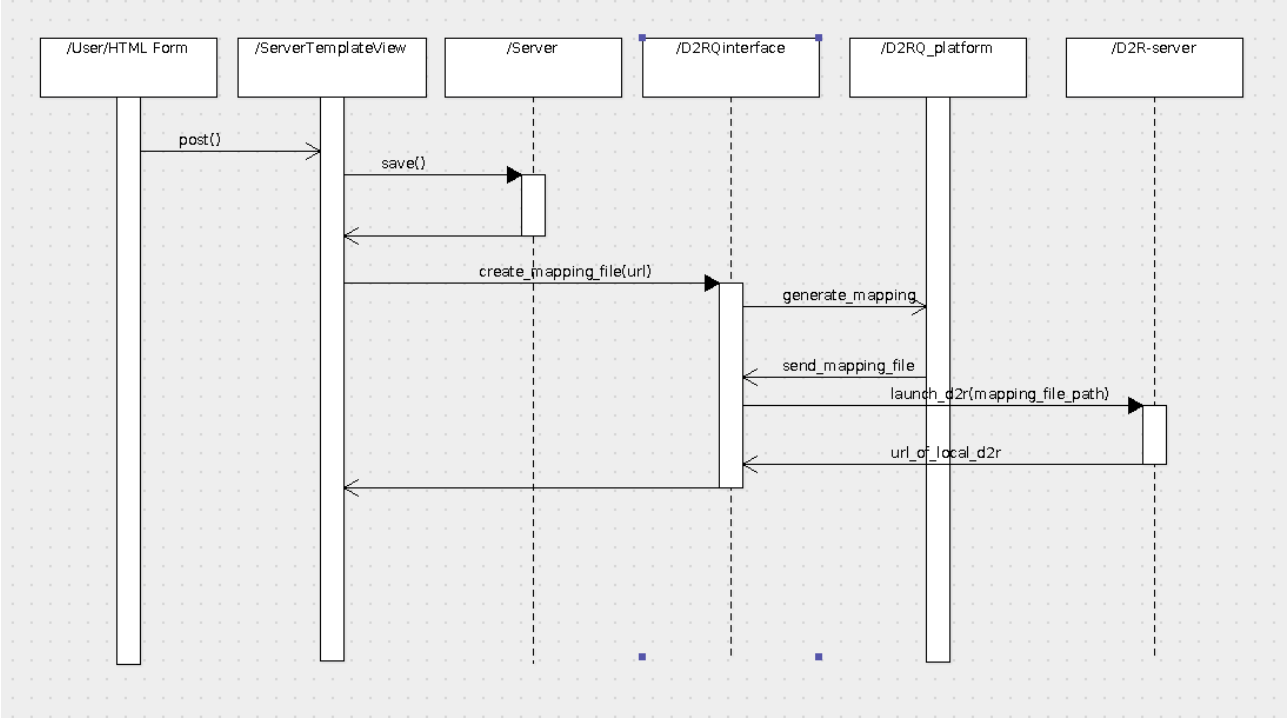
Рисунок 3.5 – Диаграмма вариантов использования

3.6.3. Диаграммы последовательностей для операций проектных классов

Диаграмма последовательности (англ. sequence diagram) — диаграмма, на которой показано взаимодействие объектов (обмен между ними сигналами и сообщениями), упорядоченное по времени, с отражением продолжительности обработки и последовательности их проявлениях [20].

Неотъемлемой частью объекта на диаграмме последовательности является линия жизни объекта. Линия жизни показывает время, в течение которого объект существует в Системе. Периоды активности объекта в момент взаимодействия показываются с помощью фокуса управления. Временная шкала на диаграмме направлена сверху вниз [20].

В ходе выполнения проекта были спроектированы диаграммы последовательностей для нескольких функций. На диаграмме 3.6 изображена диаграмма последовательности для метода launch_d2r.



Рисонок 3.6 – Диаграмма последовательности для метода launch_d2r(url)

Диаграмма описывает отправку пользователем(User) URL удаленных реляционных баз данных. Массив из URL передается на обработку в backend

веб-приложения с помощью POST запроса по протоколу HTTP. Массив обрабатывается в теле метода post() класса ServerTemplateView. В нем вызов функции save() из класса Server. Класс Server наследован от класса Model, который является встроенным классов Django и содержит необходимые методы для работы с базой данных. После сохранения вызывается метод create_mapping_file() с параметром url из класса D2RQinterface. Из этого класса происходят запросы на создание mapping-file и создание потока, в котором локально будет развернут D2R server.

На диаграмме 3.7 можно увидеть описание работы метода find_tables_name(), который вызывается из главного потока, во время работы локально развернутых серверов D2R

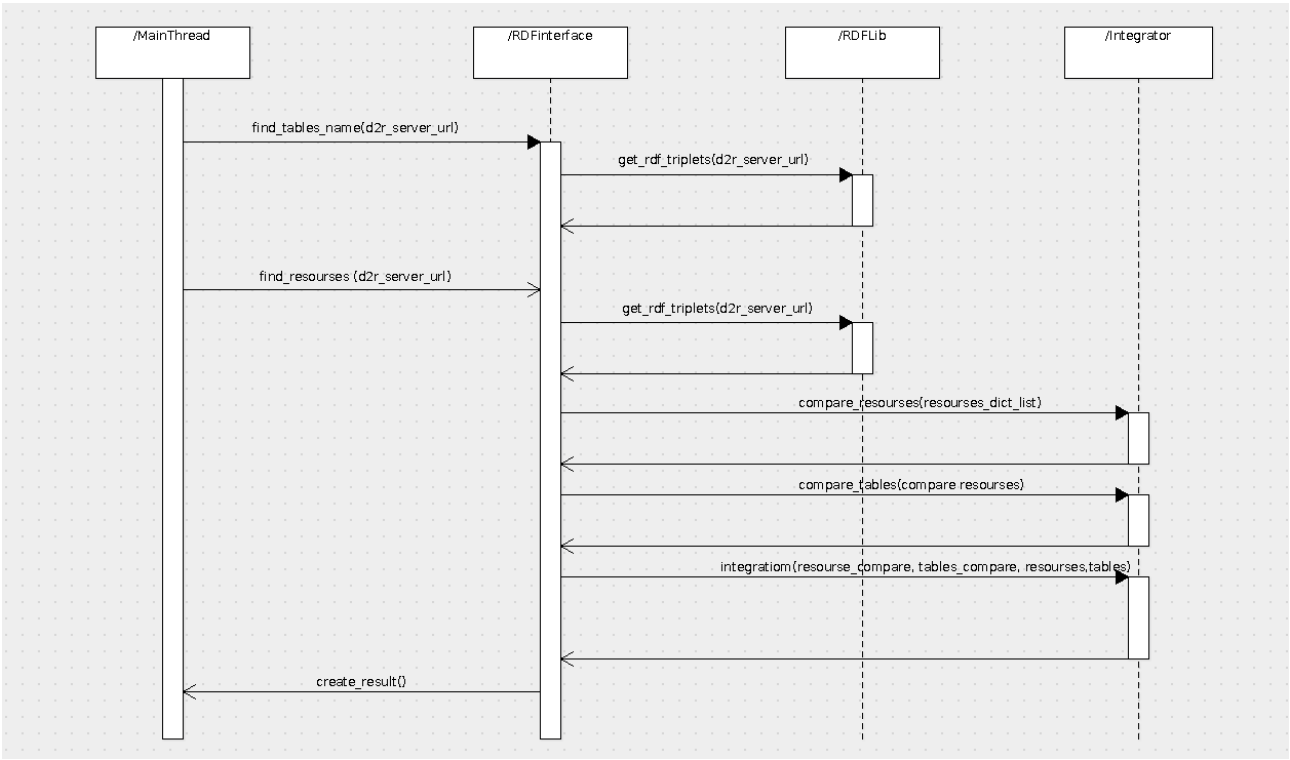


Рисунок 3.7 – Диаграмма последовательности для метода launch_d2r(url)

После вызова данного метода обработка программы начинается в классе RDFinterface, который содержит функции для нахождения названия таблиц реляционной базы данных, а также ресурсов. По очереди метод отправляет запросы в библиотеку RDF для получения полной информации об интересующих его данных. После получения списка словарей, содержащих

необходимые значения, начинается взаимодействие с классом Integrator. В этом классе сначала происходят сравнения между списками полученными из разных D2R серверов и если сравнения возвращают True, вызывается класс метод integration, в котором формируется новый RDF граф, на основе сравненных данных.

3.5. Архитектура веб-сервиса

Архитектура веб-сервиса включает в себя:

- Платформу D2RQ – как посредник адаптер
- Python/Django Framework - реализация серверная часть
- RDFLib – реализация интерфейса для обработки RDF-графов
- HTML/JavaScript – реализация клиентской части
- Ubuntu Server – платформа для развертывания

Архитектура приложения показана на схеме 3.8.

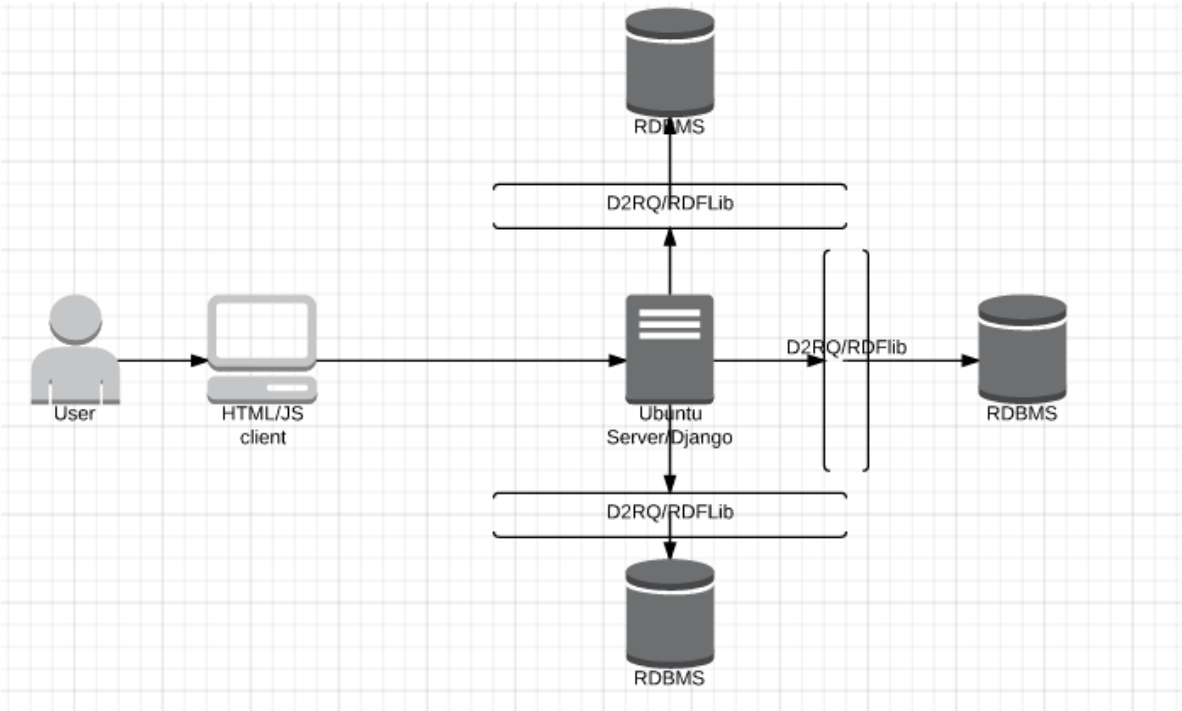


Рисунок 3.8 – Архитектура веб-сервиса для интеграции реляционных баз данных

На данном рисунке показаны следующие шаги интеграции

1. Пользователь отправляет URL реляционных баз данных через HTML формы методом POST

2. Сервер (Django) обрабатывает запрос и отправляет URL для создания mapping файлов
3. На основании mapping файлов локально запускаются D2R – серверы
4. RDFlib получает данные из D2R-серверов и классифицирует их на имена таблиц и сущности
5. Python проводит интеграцию реляционных баз данных
6. Результат интеграции обрабатывается в Django для вывода пользователю
7. Результат представлен в виде RDF и выводится на HTML страницу

4. Программная реализация системы

В результате был разработан веб-сервис реализующий интеграцию реляционных баз данных. В состав веб-сервиса входят:

- Python+Django+RDFlib – как серверная часть информационной системы
- D2RQ – как медиатор-адаптер для реляционных баз данных
- HTML+JavaScript+CSS – как клиентская часть информационной системы

4.1 Реализация серверной части веб-сервиса

Основной функционал разработанной системы находится в python-пакете Seweb. Пакет Seweb состоит из трех классов: Integrator, D2RQinterface, RDFinterface

1. Integrator предназначен для сравнения субъектов RDF триплетов и составления RDF дампа, в котором содержатся триплеты интегрированного графа в формате, доступном для добавления в RDF хранилище.
2. D2RQinterface предназначен для динамического создания и завершения
3. RDFinterface предназначен для работы с RDF графами. Он тесно связан с использованием библиотеки RDFlib.

В пакете Webinterface находятся классы и методы для обработки HTTP запросов, а также организацию ввода-вывода информации с HTML страниц.

4.2. Пример настройки платформы D2RQ

Входные параметры

Порядок работы с D2RQ платформой заключается в последовательном использовании доступных инструментов платформы. Входным параметром является реляционная база данных. Для примера, изображенного на рисунке 4.1, будет использоваться СУБД MySQL.

```
mysql> SELECT * FROM student_list;
+-------------------------+---------------+--------------+----+
| fullname                | date_of_birth | group_number | id |
+-------------------------+---------------+--------------+----+
| Rafikov Mark RAmilievich | 1993-05-01    | 8VM4V        |  1 |
| Kostin Kirill           | 1992-12-29    | 8VM4V        |  2 |
| Sanatuly Alisher        | 1992-07-26    | 8IM5V        |  3 |
+-------------------------+---------------+--------------+----+
```

Рисунок 4.1. Пример таблицы с описаниями студентов в базе данных СУБД MySQL

2. Generate - mapping

Для разворачивания D2R сервера необходимо создать mapping файл реляционной базы данных. Для этого используется инструмент D2RQ generate-mapping. Во входных необходимых параметрах он принимает JDBC Driver с указанием места хранения базы данных, логин и пароль для работы с ней. В выходных параметрах следует указать имя создаваемого файла отображения. На рисунке 4.2. показан вариант создания mapping -файла

Рисунок 4.2. Создание mapping файла

В выходном файле каждая таблица базы данных представлена как отдельный RDFS класс, который основан на имени таблицы, а его свойства — это колоны в таблице.

Таблица 4.2.3 Пример содержания mapping файла

```
@prefix map: <#> .
@prefix db: <> .
@prefix vocab: <vocab/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix d2rq: <http://www.wiwiss.fu-berlin.de/suhl/bizer/D2RQ/0.1#> .
@prefix jdbc: <http://d2rq.org/terms/jdbc/> .

map:database a d2rq:Database;
        d2rq:jdbcDriver "com.mysql.jdbc.Driver";
        d2rq:jdbcDSN "jdbc:mysql://localhost/osu";
        d2rq:username "root";
        d2rq:password "Doshirak24";
        jdbc:autoReconnect "true";
        jdbc:zeroDateTimeBehavior "convertToNull";
```

```
        .


# Table student_list
map:student_list a d2rq:ClassMap;
        d2rq:dataStorage map:database;
        d2rq:uriPattern "student_list/@@student_list.id@@";
        d2rq:class vocab:student_list;
        d2rq:classDefinitionLabel "student_list";
        .
map:student_list__label a d2rq:PropertyBridge;
        d2rq:belongsToClassMap map:student_list;
        d2rq:property rdfs:label;
        d2rq:pattern "student_list #@@student_list.id@@";
        .
map:student_list_fullname a d2rq:PropertyBridge;
        d2rq:belongsToClassMap map:student_list;
        d2rq:property vocab:student_list_fullname;
        d2rq:propertyDefinitionLabel "student_list fullname";
        d2rq:column "student_list.fullname";
        .
map:student_list_date_of_birth a d2rq:PropertyBridge;
        d2rq:belongsToClassMap map:student_list;
        d2rq:property vocab:student_list_date_of_birth;
        d2rq:propertyDefinitionLabel "student_list date_of_birth";
        d2rq:column "student_list.date_of_birth";
        d2rq:datatype xsd:date;
        .
```

```
map:student_list_group_number a d2rq:PropertyBridge;
        d2rq:belongsToClassMap map:student_list;
        d2rq:property vocab:student_list_group_number;
        d2rq:propertyDefinitionLabel "student_list group_number";
        d2rq:column "student_list.group_number";
        .
map:student_list_id a d2rq:PropertyBridge;
        d2rq:belongsToClassMap map:student_list;
        d2rq:property vocab:student_list_id;
        d2rq:propertyDefinitionLabel "student_list id";
        d2rq:column "student_list.id";
        d2rq:datatype xsd:integer;
```

3. Запуск D2R-server

После получения mapping файла появляется возможность представить базу данных, как реляционное RDF-хранилище. Следовательно, появится возможность работать с информацией из реляционной базы данных как с RDF графом.

Для запуска сервера на локальном компьютере необходимы два входных параметра: mapping файл и номер порта, который будет задействован сервером (по умолчанию: 2020). Запущенный сервер изображен на рисунке 4.3.

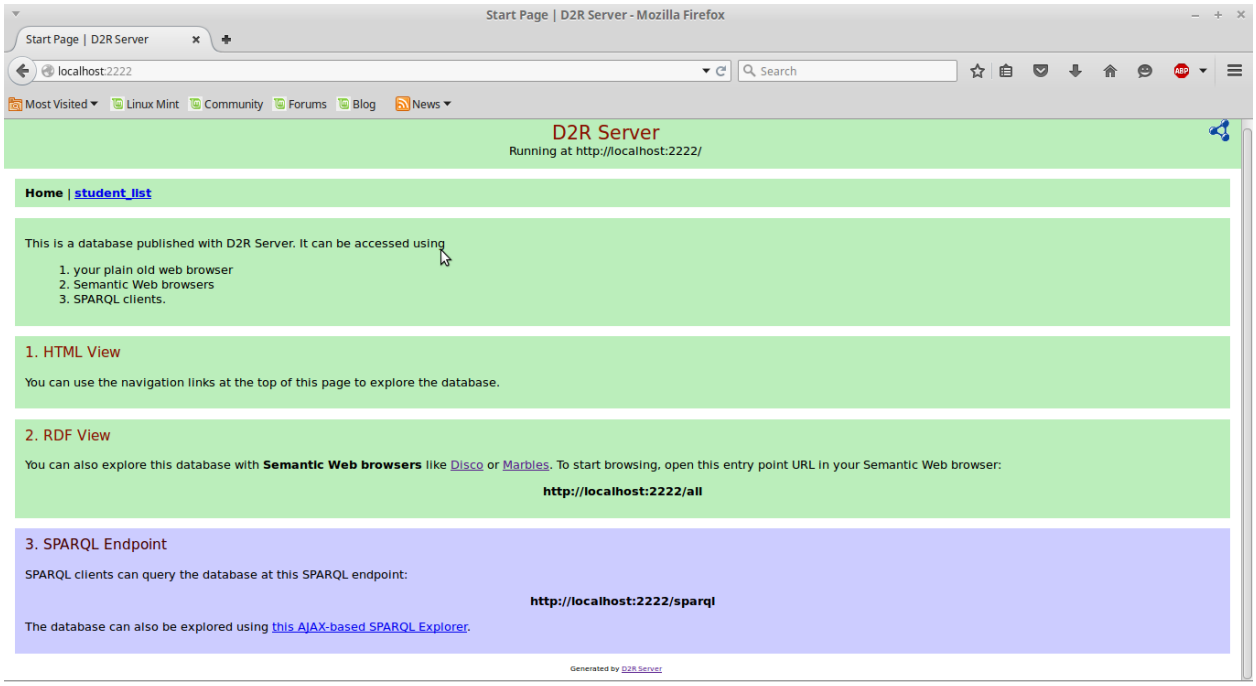


Рисунок 4.3. Запуск локального D2R-сервера

4. Dump-rdf

Представляет данные с d2r-server в файл, где вся информация расписана по триплетам <субъект><предикат><объект>. Для вызова необходимы два входных параметра : mapping файл и URI D2R сервера. Выходной параметр: файл с триплетами.

```
mark@computer ~/java_workspace/Jena/d2rq $ ./dump-rdf -f N-TRIPLE -b http://localhost:2222/ osu_mapping.ttl > osu.nt
```

Рисунок 4.4. Создание RDF дампа.

Таблица 4.2.5 Пример содержания RDF дампа

<http://localhost:2222/student_list/1> <http://localhost:2222/vocab/student_list_id> "1"^^<http://www.w3.org/2001/XMLSchema#integer> .
<http://localhost:2222/student_list/1> <http://localhost:2222/vocab/student_list_group_number> "8VM4V" .
<http://localhost:2222/student_list/1> <http://localhost:2222/vocab/student_list_date_of_birth> "1993-05-01"^^<http://www.w3.org/2001/XMLSchema#date> .
<http://localhost:2222/student_list/1> <http://localhost:2222/vocab/student_list_fullname> "Rafikov Mark RAmilievich" .
<http://localhost:2222/student_list/1> <http://www.w3.org/2000/01/rdf-schema#label> "student_list #1" .

<http://localhost:2222/student_list/1> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://localhost:2222/vocab/student_list> .
<http://localhost:2222/student_list/2> <http://localhost:2222/vocab/student_list_id>
"2"^^<http://www.w3.org/2001/XMLSchema#integer> .
<http://localhost:2222/student_list/2>
<http://localhost:2222/vocab/student_list_group_number> "8VM4V" .
<http://localhost:2222/student_list/2>
<http://localhost:2222/vocab/student_list_date_of_birth> "1992-12-
29"^^<http://www.w3.org/2001/XMLSchema#date> .
<http://localhost:2222/student_list/2> <http://localhost:2222/vocab/student_list_fullname>
"Kostin Kirill" .
<http://localhost:2222/student_list/2> <http://www.w3.org/2000/01/rdf-schema#label>
"student_list #2" .
<http://localhost:2222/student_list/2> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://localhost:2222/vocab/student_list> .
<http://localhost:2222/student_list/3> <http://localhost:2222/vocab/student_list_id>
"3"^^<http://www.w3.org/2001/XMLSchema#integer> .
<http://localhost:2222/student_list/3>
<http://localhost:2222/vocab/student_list_group_number> "8IM5V" .
<http://localhost:2222/student_list/3>
<http://localhost:2222/vocab/student_list_date_of_birth> "1992-07-
26"^^<http://www.w3.org/2001/XMLSchema#date> .
<http://localhost:2222/student_list/3> <http://localhost:2222/vocab/student_list_fullname>
"Sanatuly Alisher" .
<http://localhost:2222/student_list/3> <http://www.w3.org/2000/01/rdf-schema#label>
"student_list #3" .
<http://localhost:2222/student_list/3> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://localhost:2222/vocab/student_list> .
<http://localhost:2222/vocab/student_list_fullname> <http://www.w3.org/2000/01/rdf-
schema#label> "student_list fullname" .
<http://localhost:2222/vocab/student_list_fullname> <http://www.w3.org/1999/02/22-rdf-
syntax-ns#type> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .

<http://localhost:2222/vocab/student_list_date_of_birth> <http://www.w3.org/2000/01/rdf-schema#label> "student_list date_of_birth" .
<http://localhost:2222/vocab/student_list_date_of_birth> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
<http://localhost:2222/vocab/student_list_id> <http://www.w3.org/2000/01/rdf-schema#label> "student_list id" .
<http://localhost:2222/vocab/student_list_id> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
<http://localhost:2222/vocab/student_list> <http://www.w3.org/2000/01/rdf-schema#label> "student_list" .
<http://localhost:2222/vocab/student_list> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/2000/01/rdf-schema#Class> .
<http://localhost:2222/vocab/student_list_group_number>
<http://www.w3.org/2000/01/rdf-schema#label> "student_list group_number" .
<http://localhost:2222/vocab/student_list_group_number>
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
<http://www.w3.org/2000/01/rdf-schema#label> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property>

4.3. Реализация клиентской части веб-сервиса

Клиентская часть состоит из двух HTML страниц. На начальной странице, изображенной на рисунке 4.5, расположен блок информации о веб-сервисе и его использовании, а также HTML форма для отправки URL точек доступа для реляционных баз данных.

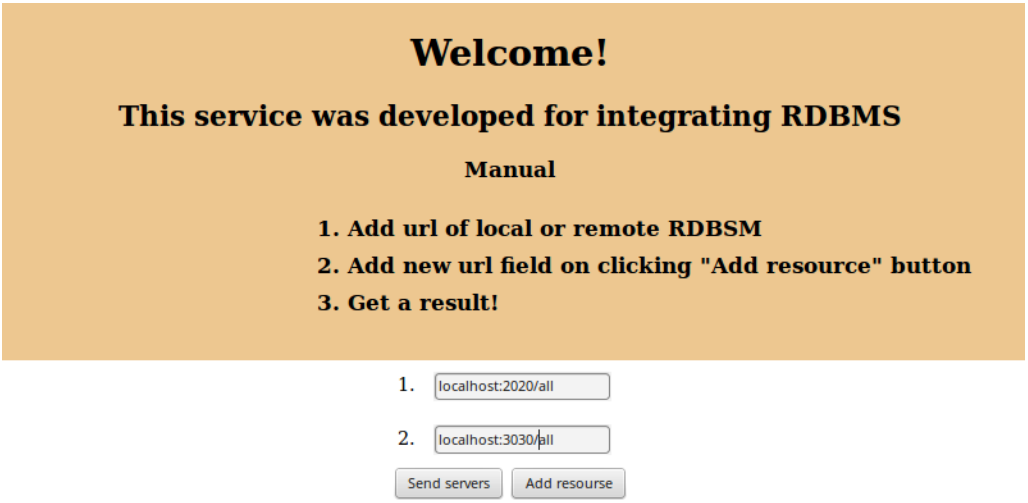


Рисунок 4.5 – Главная страница веб-сервиса

На странице с результатом выводится информация о проделанной интеграции, а также располагаются кнопки для скачивания mapping файла или RDF графа, полученных в ходе интеграции. Они могут быть полезны для добавления триплетов уже в существующие RDF хранилища.

Также в реализации клиентской часть используется JavaScript для валидации полей и отправки форм, в случае успешной валидации, а также добавления полей для новых точек доступа реляционных баз данных. Пример ошибки валидации изображен на рисунке 4.6.

Рисунок 4.6 – Пример ошибки валидации: пустая форма

5. Финансовый менеджмент, ресурсоэффективность и ресурсосбережение

5.1. Обоснование актуальности проекта.

Целью работы вносимой на защиту - проектирование и создание информационной системы интеграции реляционных баз данных используя технологию Semantic Web. И разработка программы на основе этого алгоритма.

Цель разработки - создание программы для интеграции реляционных СУБД через Semantic Web

5.2 Организация и планирование работ.

До того, чтобы приступить к разработке, необходимо рационально спланировать свою работу. Для составления планов комплекса используются сетевые и линейные методы планирования. Так как число исполнителей не превышает двух, решено было использовать линейный метод планирования.

Планирование комплекса предполагаемых работ с использованием линейного метода ведется в следующем порядке:

Составление перечня работ, необходимых для достижения поставленной задачи;

Определение участников каждой работы;

Установление продолжительности работ;

Для разработки были задействованы следующие участники:

Научный руководитель(НР)

Студент- исполнитель НИР (И)

В Таблице 5.1 приведены перечень работ и её исполнители

№	Этапы работы	Исполнители	Загрузка исполнителей
1.	Постановка целей и задач, получение исходных данных	НР	НР – 100%
2.	Составление и утверждение ТЗ	НР, И	НР-100% И-20%
3.	Подбор и изучение материалов по	НР, И	НР-50%

	тематике		И-100%
4.	Разработка календарного плана	НР, И	НР-70% И-30%
5.	Обсуждение литературы	НР, И	НР-100% И-50%
6.	Проектирование информационной системы	НР, И	НР-100% И-80%
7.	Выбор программного обеспечения	НР, И	НР-50% И-100%
8.	Реализация информационной системы	И	И-100%
9.	Оформление пояснительной записки	И	И-100%
10.	Оформление графического материала	И	И-100%
11.	Подведение итогов	НР, И	НР-60% И-100%

5.2.1. Определение трудоёмкости выполнения НИР

Для расчета трудоёмкости НИР используется экспертный способ и носит вероятностный характер, т.к. зависит от трудноучитываемых факторов. Для определения вероятных (ожидаемых) значений продолжительности работ tож применяется по усмотрению исполнителя одна из двух формул.

$$t_{ожi} = \frac{3 \cdot t_{minі} + 2 \cdot t_{maxі}}{5}, \text{ чел.-дн.,} \quad (1)$$

где $t_{ож\,i}$ – ожидаемая трудоемкость выполнения i-ой работы чел.-дн.;

$t_{min\ i}$ – минимально возможная трудоемкость выполнения заданной i-ой работы (оптимистическая оценка: в предположении наиболее благоприятного стечения обстоятельств), чел.-дн.;

$t_{max\ i}$ – максимально возможная трудоемкость выполнения заданной i-ой работы (пессимистическая оценка: в предположении наиболее неблагоприятного стечения обстоятельств), чел.-дн.

Расчет продолжительности выполнения каждого этапа в рабочих днях ($\boldsymbol{T_{РД}}$) ведется по формуле:

$$T_{РД} = \frac{t_{ож}}{K_{ВН}} \cdot K_{Д},$$

где $t_{ож}$ – продолжительность работы, дн.;

$К_{вн}$ – коэффициент выполнения работ, учитывающий влияние внешних факторов на соблюдение предварительно определенных длительностей, в частности, возможно $К_{вн}$ = 1;

$К_Д$ – коэффициент, учитывающий дополнительное время на компенсацию непредвиденных задержек и согласование работ ($К_Д$ = 1.1).

Расчет продолжительности этапа в календарных днях ведется по формуле:

$$Т_{КД} = Т_{РД} \cdot Т_К$$

где $Т_{КД}$– продолжительность выполнения этапа в календарных днях;

$Т_К$ – коэффициент календарности, позволяющий перейти от длительности работ в рабочих днях к их аналогам в календарных днях, и рассчитываемый по формуле:

$$Т_К = \frac{Т_{КАЛ}}{Т_{КАЛ} - Т_{ВД} - Т_{ПД}},$$

где $Т_{КАЛ}$ – календарные дни ($Т_{КАЛ}$ = 366);

$Т_{ВД}$ – выходные дни ($Т_{ВД}$ = 52);

$Т_{ПД}$ – праздничные дни ($Т_{ПД}$ = 10).

Таким образом:

$$Т_К = \frac{366}{366 - 52 - 10} = 1{,}204$$

Расчетные данные сведены в таблицу 5.2.

Величины трудоемкости этапов по исполнителям ТКД (данные столбцов 8 и 9 кроме итогов) позволяют построить линейный график осуществления проекта по каждому исполнителю – рисунок 1.1 приложения Б.

5.2.2. Накопленная готовность проекта

Для определения текущего состояния проекта, необходимо вычислить величину накопления готовности работы. Она показывает на сколько процентов по окончанию текущего этапа выполнен общий объём работы в целом.

Степень готовности определяется формулой:

$$СГ_i = \frac{ТР_i^Н}{ТР_{общ.}} = \frac{\sum_{k=1}^{i} ТР_k}{ТР_{общ.}} = \frac{\sum_{k=1}^{i}\sum_{j=1}^{m} ТР_{km}}{\sum_{k=1}^{I}\sum_{j=1}^{m} ТР_{km}},$$

где $ТР_{общ.}$ – общая трудоемкость проекта ($ТР_{общ.}$=129);

$ТР_i^Н$ – накопленная трудоемкость i-го этапа проекта по его завершении;

Таблица 5.3 – Нарастание технической готовности

Этап	$ТР_i$, %	$СГ_i$, %
Постановка целей и задач, получение исходных данных	1,58	1,58
Составление и утверждение ТЗ	9,48	11,06
Подбор и изучение материалов по тематике	23,84	34,90
Разработка календарного плана	4,28	39,19
Обсуждение литературы	15,73	54,92
Проектирование информационной системы	5,36	60,28
Выбор программного обеспечения	12,64	72,93
Реализация информационной системы	10,83	83,76

Оформление пояснительной записки	4,87	88,64
Оформление графического материала	3,65	92,30
Подведение итогов	7,69	100

5.3. Состав сметы затрат на выполнение проекта

Затраты на выполнение проекта рассчитываются по следующим статьям расходов с последующим суммированием :

- заработная плата;
- электроэнергия
- страховые взносы;
- прочие расходы.

5.3.2. Заработная плата

Затраты на выплату заработной платы указаны в таблице 5.4.

Таблица 5.4 Затраты на выплату заработной платы

Исполнитель	Оклад, руб./мес.	Среднедневная ставка, руб./раб.день	Затраты времени, раб.дни	Коэффициент	Фонд з/платы, руб.
НР	**33 162,87**	**1335,59**	**70**	**1,699**	**158 842,53**
И	**7 864,11**	**316,71**	**120**	**1,699**	**64 572,48**
Итого:					**223 415,02**

5.3.3 Затраты на социальный налог

Затраты на уплату страховых взносов по установленным государством нормам, которые составляют 30% от заработной платы сотрудника.

Страховые взносы включают в себя:

$С_{соц.} = С_{зп} * 0,3$

$С_{соц.} =$ **193 512,76р** *0,3= 67024,50 рублей

5.3.4 Затраты на электроэнергию

Данный вид затрат, включат в себя всё затраты, связанные с эксплуатацией персонального компьютера. Для определения часа работы оборудования используется формула:

$$С_{эл.об.} = Р_{об} \cdot t_{об} \cdot Ц_{Э},$$

где $P_{ОБ}$ – мощность, потребляемая оборудованием, кВт;

$Ц_Э$ – тариф на 1 кВт·час;

$t_{об}$ – время работы оборудования, час.

Учитывая, что общая продолжительность реализации проекта для исполнителя составляет 120 рабочих дней, а для научного руководителя – 70 рабочих дней, тарифная ставка корпусов и общежитий ТПУ и составляет 5,257, а средняя продолжительность дня 8 часов, затраты состоят.

Таблица 5.5 – Затраты на электроэнергию

Наименование оборудования	Время работы оборудования $t_{ОБ}$, час	Потребляемая мощность $P_{ОБ}$, кВт	Затраты $Э_{ОБ}$, руб.
Персональный компьютер И	864	0,3	1362,61
Персональный компьютер И	410,4	0,3	647,24
Итого:			**2009,85**

5.3.5 Расчет амортизационных расходов

В статье «Амортизационные отчисления» рассчитывается амортизация используемого оборудования за время выполнения проекта.

Используется формула:

$$С_{АМ} = \frac{Н_А * Ц_{ОБ} * t_{рф} * n}{F_Д},$$

где $Н_А$ – годовая норма амортизации единицы оборудования;

$Ц_{ОБ}$ – балансовая стоимость единицы оборудования;

$F_Д$ – действительный годовой фонд времени работы соответствующего оборудования;

$t_{рф}$ – фактическое время работы оборудования в ходе выполнения проекта, учитывается исполнителем проекта;

n – число задействованных однотипных единиц оборудования.

Расчет амортизации используемого оборудования

Таблица 5.6 Расчет затрат на электроэнергию

Исполнитель	$Н_А$	$Ц_{ОБ}$	$F_Д$	$t_{рф}$	n	$С_{АМ}$
НР	0,4	29000	864	2384	1	1362,61
И	0,4	45000	410,4	2384	1	647,24
Итого						2009,85

5.3.6 Расчет прочих расходов

В статье «Прочие расходы» отражены расходы на выполнение проекта, которые не учтены в предыдущих статьях, их следует принять равными 10% от суммы всех предыдущих расходов, т.е.

$$Спроч. = (Сзп + Ссоц + Сэл.об. + Сам) \cdot 0,1$$

Таким образом, прочие расходы составляют:

Спроч. = 30 207,15руб

5.3.7. Общая себестоимость разработки

Все вышеперечисленные затраты включаются в смету, которая приведена в таблице 5.7.

Таблица 5.7 - Смета затрат на разработку

Статья затрат	**Условное обозначение**	Сумма
Основная заработная плата	$C_{зп}$	223 415,02
Отчисления в социальные фонды	$C_{соц}$	67024,50
Расходы на электроэнергию	$C_{эл.}$	2009,85
Амортизационные отчисления	$C_{ам}$	9622,14
Прочие расходы	$C_{проч}$	30 207,15
Итого:		332278,68

Затраты на разработку составили С = 332 278,68руб

Прибыль = затраты на разработку *5%= 315 709,82* 0,05 = 16 613,93р.

Цена проекта с учетом НДС= (332 278,68 руб. + 16 613,93руб.) * 0,18 = 62 800,66 руб.

Итоговая цена разработки НИР:

ЦНИР(КР) = 332 278,68руб. + 16 613,93руб + 62 800руб = 411 693,28руб.

5.4. Определение уровня НИР

Дадим оценку организационной эффективности разработки, используя таблицу 5.8.

Таблица 5.8 - Оценка организационной эффективности разработки

Признаки научно-технического эффекта НИР	**Характеристика признака НИР**	**Ri**	**Выбранный бал**
Уровень новизны	По-новому объясняются те же факты, закономерности, новые понятия дополняют ранее полученные ре-зультаты	0,4	7
Теоретический уровень	Разработка способа (алгоритм, программа мероприятий, устройство, вещество и т.п.)	0,1	6
Возможность реализации	Время реализации в течение первых лет	0,5	10

Отсюда интегральный показатель научно-технического уровня для проекта составляет:

$$I_{\text{нту}} = 0,4*7 + 0,1*6 + 0,5*10 = 8,4$$

Таким образом, данный проект имеет высокий уровень научно-технического эффекта.

5.5. Оценка экономической эффективности проекта

Работа не является законченным проектом, выполненная часть является составляющей большего проекта. Отсутствуют четкие перспективы его внедрения в практику, то оценка соответствующего экономического эффекта, а, следовательно, эффективности в рамках представленной работы, невозможна в виду отсутствия необходимых данных.

6. Социальная ответственность

Разработанный программный продукт служит для обработки цифровой информации (изображений и видео) посредством использования персонального компьютера. Таким образом, данный проект подразумевает взаимодействие пользователя с ПЭВМ, а также соответствующими периферийными устройствами, подключаемыми к компьютеру.

Принимая во внимание назначение созданного проекта, а также существенное нарастание числа пользователей персональных компьютеров, характерное для современного периода развития компьютерных и интернет-технологий, обеспечение безопасности пользователей ПК является крайне актуальным вопросом.

Данный раздел посвящен анализу опасных и вредных факторов применительно рассматриваемому виду производственной деятельности [31], разработке мер по защите в чрезвычайных ситуациях, а также описанию правовых и организационных вопросов обеспечения безопасности. Таким образом, в разделе решаются следующие задачи:

- анализ вредных факторов проектируемой производственной среды;
- анализ опасных факторов проектируемой среды;
- охрана окружающей среды;
- защита в чрезвычайных ситуациях;
- правовые и организационные вопросы обеспечения безопасности.

Прежде чем перейти к рассмотрению всех указанных задач, необходимо произвести оценку рабочего помещения на соответствие санитарно-техническим условиям труда

Характеристика рабочего помещения

Стандартное офисное рабочее место обладает естественным и искусственным освещением и имеет следующие параметры: длина = 4 м, ширина = 6 м, высота = 3 м. Общая площадь помещения составляет 24 м^2, а

объем = 72 м^3. Помещение оснащено 3 компьютерами, что означает возможность работы 3 человек одновременно. Исходя из параметров помещения и количества рабочих мест, получаем следующие параметры на одно рабочее место: площадь = 8 м^2, объем = 24 м^3. Таким образом, помещение удовлетворяет в СанПиН 2.2.2/2.4.1340-03 нормам, согласно которым площадь на одно рабочее место пользователей ПЭВМ (с жидкокристаллическими, плоскими экранами) должна быть не менее 4,5 м^{2,} а объем – не менее 21 м^3

6.1. Производственная безопасность.

6.1.1 Анализ вредных факторов проектируемой производственной среды

Вредными считаются производственные факторы, воздействие которых может повлечь за собой заболеваниям работника или нарушению состояния его здоровья в зависимости от характера воздействия. Применительно к данной работе в качестве вредных факторов рассматриваются такие как неблагоприятный микроклимат, повышенный уровень шума, вибрации, плохое освещение, а также неблагоприятный аэроионный состав воздуха[32].

6.1.1.1 Неблагоприятный микроклимат

Под микроклиматом помещения понимают совокупность его физических факторов, которые оказывают влияние на тепловой обмен организма и здоровье человека. В соответствие с СанПиН 2.2.4.548-96 микроклимат в помещении характеризуется следующими показателями:

- температура воздуха;
- температура поверхностей;
- относительная влажность воздуха;
- интенсивность теплового облучения;
- скорость движения воздуха.

Все перечисленные параметры оказывают огромное влияние на функционирование организма [32]. Например, слишком высокая или низкая температура воздуха увеличивает и уменьшает его теплоотдачу, что в том и

другом случае является неестественным состоянием; слишком низкая (менее 20%) влажность воздуха может привести к пересыханию слизистых оболочек, а слишком высокая влажность (более 80%) затрудняет процесс терморегуляции [33]..

Для помещений с компьютерами актуально соблюдение всех перечисленных норм, так как наличие в ПЭВМ различных излучающий и нагревательных элементов способно значительно повлиять на составляющие микроклимата. В таблице 6.1 представлены оптимальные величины показателей микроклимата (согласно СанПиН 2.2.4.548-96). В качестве примера рассматривается категория работ Iа. «К категории Iа относятся работы с интенсивностью энергозатрат до 120 ккал/ч (до 139 Вт), производимые сидя и сопровождающиеся незначительным физическим напряжением (ряд профессий на предприятиях точного приборо- и машиностроения, на часовом, швейном производствах, в сфере управления и т. п.)» Данное описание характерно для работы с персональным компьютером.

Таблица 6.1 – Оптимальные величины показателей микроклимата на рабочих местах производственных помещений

Период года	Категория работ по уровню энергозатрат, Вт	Температура воздуха, °С	Температура поверхностей, °С	Относительная влажность воздуха, %	Скорость движения воздуха, м/с
Холодный	Iа (до 139)	22-24	21-25	60-40	0,1
Теплый	Iа (до 139)	23-25	22-26	60-40	0,1

В таблице 6.2 приведены допустимые величины показателей микроклимата для категории Iа. Допустимые величины устанавливаются только в тех случаях, когда наблюдаются кратковременные отклонения от оптимальных величин.

Таблица 6.2 – Допустимые величины показателей микроклимата

Период года	Категория работ по	Температура воздуха, °С	Температура	Относительная	Скорость движения воздуха, м/с

	уровню энергозатрат, Вт	диапазон ниже оптимальных величин	диапазон выше оптимальных величин	поверхностей, °С	влажность воздуха, %	для диапазона температур воздуха ниже оптимальных величин, не более	для диапазона температур воздуха выше оптимальных величин, не более
Холодный	Ia (до 139)	20,0-21,9	24,1-25,0	19,0-26,0	15-75	0,1	0,1
Теплый	Ia (до 139)	21,0-22,9	25,1-28,0	20,0-29,0	15-75	0,1	0,2

6.1.1.2 Повышенный уровень шумов, вибраций

При работе с компьютером пользователь на протяжении всего времени работы находится под воздействием шумов и вибраций, создаваемых различными компонентами ПЭВМ. К источникам шума можно отнести систему охлаждения компьютера, работу центрального процессора, а также шумы от принтера и других подключаемых устройств. Согласно требованиям СанПиН 2.2.4/2.1.8.562-96 уровень шума на рабочем месте пользователя персонального компьютера не должен превышать 50 дБА. Согласно СН 2.2.4/2.1.8.566-96 уровень вибрации на рабочем месте пользователя (категория 3, тип «в») должен соответствовать показателям, представленным в таблице 6.3.

Таблица 6.3 - Предельно допустимые вибрации рабочих мест категории 3 технологической группы типа "в" [2]

Среднегеометрические частоты полос, Гц	Предельно допустимые значения по осям Xo, Yo, Zo							
	виброускорения				виброскорости			
	м/с2		дБ		м/с 10-2		дБ	
	1/3 окт	1/1 окт	1/3 окт	1/1 окт	1/3 окт	1/1 окт	1/3 окт	1/1 окт
1,6	0,0130		82		0,130		88	
2,0	0,0110	0,020	81	86	0,089	0,180	85	91
2,5	0,0100		80		0,063		82	
3,15	0,0089		79		0,045		79	
4,0	0,0079	0,014	78	83	0,032	0,063	76	82
5,0	0,0079		78		0,025		74	
6,3	0,0079		78		0,020		72	
8,0	0,0079	0,014	78	83	0,016	0,032	70	76
10,0	0,0100		80		0,016		70	
12,5	0,0130		82		0,016		70	
16,0	0,0160	0,028	84	89	0,016	0,028	70	75
20,0	0,0200		86		0,016		70	
25,0	0,0250		88		0,016		70	
31,5	0,0320	0,056	90	95	0,016	0,028	70	75
40,0	0,0400		92		0,016		70	
50,0	0,0500		94		0,016		70	
63,0	0,0630	0,110	96	101	0,016	0,028	70	75
80,0	0,0790		98		0,016		70	
Корректированные и эквивалентные корректированные		0,014		83		0,028		75

Среднегеометрические частоты полос, Гц	Предельно допустимые значения по осям Xo, Yo, Zo							
	виброускорения				виброскорости			
	м/с2		дБ		м/с 10-2		дБ	
	1/3 окт	1/1 окт	1/3 окт	1/1 окт	1/3 окт	1/1 окт	1/3 окт	1/1 окт
значения и их уровни								

Снизить уровень шумов в помещениях можно за счет использования звукопоглощающих материалов для отделки стен и потолков. «Дополнительный звукопоглощающий эффект создают однотонные занавески из плотной ткани, повешенные в складку на расстоянии 15-20 см от ограждения. Ширина занавески должна быть в 2 раза больше ширины окна».

6.1.1.3 Недостаточная освещенность рабочего места

Освещенность помещения является важным фактором в условиях постоянной работе с компьютером. Так как органы зрения испытывают постоянное воздействие от экрана монитора, необходимо, чтобы окружающее освещение не было дополнительным источником негативного воздействия на глаза человека.

Недостаточное освещение рабочего места повышает утомляемость работника, снижает внимательность, уменьшает производительность труда и способствует развитию близорукости. Уже через 40-45 минут непрерывной работы за компьютером, у пользователя могут появиться первые признаки дискомфорта, а через 2 часа происходит значительное ухудшение зрительных функций [34].

Согласно СП 52.13330.2011 для административных помещений, оборудованных компьютером, нормативные показатели освещенности компьютерных залов имеют вид, представленный в таблице 6.4.

Таблица 6.4 - Нормативные показатели освещения компьютерных залов

1			2	3
Плоскость нормирования освещенности и КЕО, высота плоскости над полом, м			Вертикальная–1,2 (на экране дисплея)	Горизонтальная–0,8 (на рабочих столах)
Разряд и подразряд зрительной работы			Б–2	А–2
Искусственное освещение	Освещенность рабочих поверхностей, лк	комбинированное освещение	-	500/300
		общее освещение	200	400
1			2	3
Искусственное освещение	Освещенность рабочих поверхностей, лк	Объединенный показатель дискомфорта UGR, не более	-	14
		Коэффициент пульсации освещенности, не более	-	10
Естественное освещение	КЕО ЕН, %	верхнее или комбинированное освещение	-	3,5
		боковое освещение	-	1,2
Совмещенное освещение	КЕО ЕН, %	верхнее или комбинированное освещение	-	2,1
		боковое освещение	-	0,7

В СанПиН 2.2.2/2.4.1340-03 описаны основные санитарно-эпидемиологические требования к освещению на рабочих местах, оборудованных компьютерами. Приведем некоторые из них:

- Освещенность на поверхности стола в зоне размещения рабочего документа должна быть 300 лк.

- Освещение не должно создавать бликов на поверхности экрана. Освещенность поверхности экрана не должна быть более 300 лк
- В качестве источников света при искусственном освещении следует применять преимущественно люминесцентные лампы типа ЛБ и компактные люминесцентные лампы (*КЛЛ*).

6.1.1.4 Повышенный уровень электромагнитного излучения

Источниками электромагнитного поля в аудитории являются мониторы, клавиатуры, процессоры, блоки питания, свитчи, маршрутизаторы. Предельно-допустимые нормы указаны в таблице 6.5.

Таблица 6.5. Допустимые уровни ЭМП

Напряженность электрического поля	
в диапазоне частот 5 Гц — 2 кГц, E1	25 В/м
в диапазоне частот 2 кГц — 400 кГц, E2	2,5 В/м
Плотность магнитного потока	
в диапазоне частот 5 Гц — 2 кГц, B1	250 нТл
в диапазоне частот 2 кГц — 400 кГц, B2	25 нТл

Для монитора были получены следующие показатели: E1 = 6.3 В/м, E2 = 0.3 В/м, B1 = 44.2 В/м, B2 = 7.66 В/м. Очевидно, что данные показатели значительно меньше требуемых норм и, соответственно, излучение минимально. Минимальные значения выявлены и при анализе другой техники. Единственное отклонение возникает при включении маршрутизатора: магнитная составляющая B1 = 520 В/м из-за наличия в блоке питания маршрутизатора трансформатора. Возможным решением является экранирование трансформатора.

Мощность рентгеновского излучения на расстоянии 0.05 м от экрана монитора не превышает 1 мкЗв/ч, что соответствует нормам.

В аудитории имеется электропроводка напряжением 220 вольт, предназначенная для питания вычислительной техники и освещения. Электропроводка тщательно изолирована и защищена, поэтому вероятность возгорания сведена к минимуму. Магнитное поле вблизи электропроводки (3 см) составляет 70 мкТл, что соответствует нормам.

Мощных источников магнитного поля в аудитории нет. Обеспечивается надежное заземление, подводимое к каждому рабочему месту.

6.1.1.5 Психофизиологическое воздействие

Применительно к работе за компьютером под психофизическими факторами понимают: монотонный режим работы, напряжение зрения, памяти, внимания, эмоциональные перегрузки.

Основным фактором, влияющим на нервную систему пользователя ПК, является большой поток информации, который он вынужден воспринимать.

После непрерывной работы за компьютером к концу рабочего дня пользователь может испытывать переутомление глаз, головную боль, боль в мышцах спины, а также в области шеи.

Одним из способов предупреждения преждевременной утомляемости пользователей ПЭВМ является организовывать рабочую смену путем чередования работ с использованием ПЭВМ и без него. При высоком уровне напряженности работы рекомендуется психологическая разгрузка в специально оборудованных помещениях.

Помимо организации перерывов на протяжении рабочего дня, крайне важна правильная конструкция рабочего места работника. Данные вопросы подробно рассматриваются в разделе 5.5

6.1.2 Анализ опасных факторов производственной среды

Опасным считается производственный фактор, воздействие которого может привести к травме работника или даже повлечь за собой его смерть. С учетом рассматриваемой производственной среды, в данном разделе рассматриваются такие факторы как механические и термические опасности, электробезопасность и пожаробезопасность.

6.1.2.1 Механические опасности

Рабочее место пользователя ПК оснащено достаточно большим количеством компонентов компьютера, поэтому во избежание получения травм от падения каких-либо предметов, все оборудование должно быть размещено на устойчивых поверхностях. Кроме того, все составляющие рабочего места должны учитывать физические особенности человека и не препятствовать его свободному движению. Компьютерные классы, как правило, характеризуются достаточно высокой плотность размещения техники, которая в свою очередь подразумевает наличие множества проводов. В целях обеспечения безопасности рабочего места все провода и соединительные элементы должны быть размещены таким образом, чтобы не препятствовать перемещению пользователя по всему пространству помещения.

6.1.2.2 Электробезопасность

Компьютер содержит большое количество компонентов, питающихся от источников тока, таких как монитор, системный блок, принтер, клавиатура, мышь. Все перечисленное, а также множество соединительных проводов компонент представляют потенциальную угрозу воздействия тока на пользователя. Поэтому, во избежание поражения током, крайне важно соблюдать основные правила по электробезопасности при работе с ПЭВМ.

Прежде всего, перед началом работы нужно убедиться в целостности вилки и провода электропитания, а также в отсутствии повреждений компонент ПЭВМ. При обнаружении любого вида неисправностей необходимо немедленно обратиться к администрации или уполномоченному техническому персоналу.

Во избежание поражения электрическим током запрещается:

- прикасаться задней панели системного блока, а также тыльной стороне дисплея компьютера;
- работать за компьютером во влажной одежде или влажными руками;
- вытирать пыль с компьютера во включенном состоянии;

- использовать жидкие или аэрозольные чистящие средства для осуществления чистки компьютера;
- касаться одновременно каких-либо трубопроводов, батарей отопления, металлических конструкций, соединенных с землей (при пользовании электроприборами);
- класть посторонние предметы на средства вычислительной техники, а также периферийные устройствам [37, 39].

Необходимо соблюдать также следующие меры по поддержке электробезопасности:

- Постоянно контролировать надежность соединения контактов проводов
- Подключать такие компоненты компьютера, как монитор и дисплей через согласующее устройство (сетевой фильтр)
- Не ставить системный блок в зоне повышенно влажности и запыленности;
- Производить ежедневную влажную уборку помещения во избежании запыленности воздуха [37, 39];

Нельзя не принимать во внимание также статического электричества, образующегося в процессе работы компьютера на корпусе монитора, клавиатуры и системного блока. Хотя данный вид электричества не представляет опасности для человека, его действие может привести к выходу из строя компьютера. «Для снижения величин токов статического электричества используются нейтрализаторы, местное и общее увлажнение воздуха, использование покрытия полов с антистатической пропиткой»

6.1.2.3 Пожаробезопасность

Согласно Нормам пожарной безопасности НТБ 105-95, помещения с ПЭВМ относятся к категории В (пожароопасные)

Помещения, в которых размещены компьютеры, как правила характеризуются сравнительно небольшой площадью. Основными источниками возгорания в таких помещениях могут стать устройства электропитания,

электрические схемы от ЭВМ, различные приборы технического обслуживания, у которых в случае нарушений их работы может возникать перегревание элементов.

Кроме того, для современных компьютеров характерна достаточно высокая плотность микросхем и высокая близость расположения соединительных проводов и кабелей. При протекании по данным компонентам электрического тока, образуется большое количество теплоты, которое может привести к оплавлению изоляции или других составляющих, и, как следствие, к возгоранию техники и всего помещения. Таким образом тесное взаимодействие компонентов и внутренних схем компьютера представляют дополнительную пожарную безопасность.

В качестве профилактических мероприятий для обеспечения пожарной безопасности следует использовать скрытую электросеть, надежные розетки из пожаробезопасных материалов. Нужно регулярно делать очистку внутренних частей ПЭВМ и другого оборудования от пыли, располагать компьютеры на отдельных столах. Вентиляционные отверстия в электроаппаратуре, предназначенные для охлаждения, запрещается перекрывать другим оборудованием или любыми другими твердыми и мягкими предметами. После окончания работы необходимо обесточить все средства вычислительной техники и периферийное оборудование. В случае непрерывного производственного процесса необходимо оставить включенными только необходимое оборудование.

При возникновении пожароопасной ситуации или пожара персонал должен немедленно принять необходимые меры для его ликвидации, одновременно оповестить о пожаре администрацию.

Помещения с электроборудованием должны быть оснащены огнетушителями типа ОУ-2 или ОУБ-3, а также надежной системой пожарной сигнализации и планов эвакуации для персонала.

6.1.3. Для поддержания нормальных значений параметров

микроклимата

На рабочих метах рекомендуется оснащать их системами отопления, вентиляции и кондиционирования воздуха. Также, в некоторых случаях, целесообразно обеспечить питьевое водоснабжение. В помещениях для работы с ПЭВМ должна производиться ежедневная влажная уборка, а также систематическое проветривание после каждого часа работы [35].

Для защиты операторов ПЭВМ от негативного воздействия электромагнитных полей в первую очередь необходимо, чтобы используемая техника удовлетворяла нормам и правилам сертификации. При работе с ПЭВМ установлены регламентированные перерывы, а также иногда предусмотрено использование экранов и фильтров в целях защиты оператора [37].

Для создания и поддержания благоприятных условий освещения для операторов ПЭВМ, их рабочие места должны соответствовать санитарно-эпидемиологическим правилам СанПиН 2.2.2/2.4.1340-03. Рабочее помещение должно иметь естественное и искусственное освещение, соответствующее показателям, представленным в таблице **Ошибка! Источник ссылки не найден.**. Для рассеивания естественного освещения следует использовать жалюзи на окнах рабочих помещений. В качестве источников искусственного освещения должны быть использованы люминесцентные лампы, лампы накаливания – для местного освещения [38].

Для предупреждения преждевременной утомляемости пользователей ПЭВМ рекомендуется организовывать рабочую смену путем чередования работ с использованием ПЭВМ и без него. В случаях, когда характер работы требует постоянного взаимодействия с компьютером (работа программиста-разработчика) с напряжением внимания и сосредоточенности, при исключении возможности периодического переключения на другие виды трудовой деятельности, не связанные с ПЭВМ, рекомендуется организация перерывов на 10–15 мин. через каждые 45–60 мин. работы. При высоком уровне напряженности работы рекомендуется психологическая разгрузка в специально оборудованных помещениях [37].

К мероприятиям по предотвращению возможности поражения электрическим током относятся:

- При производстве монтажных работ необходимо использовать только исправный инструмент, аттестованный службой КИПиА;
- С целью защиты от поражения электрическим током, возникающим между корпусом приборов и инструментом при пробое сетевого напряжения на корпус, корпуса приборов и инструментов должны быть заземлены;
- При включенном сетевом напряжении работы на задней панели должны быть запрещены;
- Все работы по устранению неисправностей должен производить квалифицированный персонал;
- Необходимо постоянно следить за исправностью электропроводки [37, 39].

Для профилактики организации действий при пожаре должен проводиться следующий комплекс организационных мер: должны обеспечиваться регулярные проверки пожарной сигнализации, первичных средств пожаротушения; должен проводиться инструктаж и тренировки по действиям в случае пожара; не должны загромождаться или блокироваться пожарные выходы; должны выполняться правила техники безопасности и технической эксплуатации электроустановок; во всех служебных помещениях должны быть установлены «Планы эвакуации людей при пожаре и других ЧС», регламентирующие действия персонала при возникновении пожара.

Для предотвращения пожара помещение с ПЭВМ должно быть оборудовано первичными средствами пожаротушения: углекислотными огнетушителями типа ОУ-2 или ОУ-5; пожарной сигнализацией, а также, в некоторых случаях, автоматической установкой объемного газового пожаротушения [40].

6.2 Охрана окружающей среды

Компьютерная индустрия, как им многие другие сферы человеческой деятельности обладает негативным воздействием на окружающую среду.

Одной из важнейших проблем, связанных с данной индустрией является проблема загрязнения окружающей среды на стадии разработки компьютерного оборудования, а также в момент его утилизации. По мнению исследователей из ООН, на изготовление одного среднестатистического персонального компьютера, общий вес различных используемых химикатов в 10 раз превосходит вес окончательного продукта[6]. Кроме того, существует проблема утилизации вышедших из строя или из обращения компьютеров и их компонентов[41].

Большинство современных компаний в настоящее время стремится свести к минимуму использование токсичных материалов при производстве компьютеров, все же данная проблема существует. Кроме того, не все устаревшие модели компьютеров выведены из оборота, а, как известно, при производстве подобных моделей использовались различные тяжелые металлы[42].

Более того, несмотря на то, что в современных компьютерах потребление электроэнергии сводится к минимуму, затраты электроэнергии, а значит и природных ресурсов, на этапе их разработки огромны по своим масштабам[42-44].

Таким образом, необходим ряд мер, осуществляющий регулирование данных вопросов на законодательном уровне.

6.3 Защита в чрезвычайных ситуациях

Чрезвычайные ситуации (ЧС) имеют следующую классификацию:

1) ЧС техногенного характера: пожар, внезапное обрушение здания
2) ЧС природного характера: землетрясения, бури, ураганы, наводнения, лесные пожары
3) ЧС биологические: эпидемии, эпизоотии, эпифитотии.

4) ЧС социальные: межнациональные конфликты, грабежи, терроризм, войны, геноцид
5) Антропогенные ЧС: следствия ошибочных действий людей[4]

Применительно к рассматриваемой рабочей среде наиболее характерно возникновение ЧС техногенного характера, однако при этом не исключается вероятность возникновения любой из перечисленных категорий. Наиболее типичной ЧС техногенного характера для рабочего помещения, оборудованного ПЭВМ, является пожар. Поэтому рассмотрим основные меры по предотвращению пожара, а также основные действия в случае его возникновения[45].

Меры предосторожности.

Данные меры, как правило, заложены в основу инструктажа по пожаробезопасности каждого сотрудника. Соблюдение данных мер обязательно при нахождении рабочем месте.

1) Запрещается использовать электроприборы в условиях, не соответствующих требованиям инструкций изготовителей, или имеющие неисправности, которые в соответствии с инструкцией по эксплуатации могут привести к пожару, а также эксплуатировать электропровода и кабели с поврежденной или потерявшей защитные свойства изоляцией.
2) При длительных перерывах более 1 часа или при уходе с работы необходимо выключать персональный компьютер и другие электроприборы из сети электропитания
3) Покидая рабочее место, сотрудник должен убедиться в отсутствии источников возможного возгорания[34].

Меры по повышению устойчивости объекта к возникновению ЧС

Повышение устойчивости достигается за счет проведения соответствующих организационно-технических мероприятий, в том числе применения правильной внутренней планировки и застройки территории и

подготовки персонала к работе в ЧС. Здания для вычислительных центорв и части здания другого назначения, в которых предусмотрено размещение ЭВМ, должны быть первой и второй степени огнестойкости. Для изготовления строительных конструкций используются, как правило, кирпич, железобетон, стекло, металл и другие негорючие материалы. Применение дерева должно быть ограничено, а в случае использования необходимо пропитывать его огнезащитными составами[46].

Действия в случае возникновения ЧС

При обнаружении пожара работнику необходимо:

- немедленно прекратить работу и вызвать пожарную охрану по телефону «01», сообщив при этом адрес, место возникновения пожара и свою фамилию;
- принять по возможности меры по эвакуации людей и материальных ценностей .
- отключить от сети закрепленное за ним электрооборудование;
- приступить к тушению пожара имеющимися средствами пожаротушения;
- сообщить непосредственному или вышестоящему начальнику и оповестить окружающих сотрудников;
- при общем сигнале опасности покинуть здание.

6.4. Правовые и организационные вопросы обеспечения безопасности

6.4.1 Требования к организации рабочих мест пользователей ПЭВМ

Согласно СанПиН 2.2.2/2.4.1340-03 при организации рабочего места пользователя компьютера необходимо соблюдать следующие требования:

- Расстояние между рабочими столами с видеомониторами должно составлять не менее 2 м в направлении тыльной стороны монитора, и не менее 1,2 м между боковыми поверхностями мониторов.

- Расстояние от монитора до глаз пользователя должно быть не менее 600-700 мм, при определенном размере шрифта допускается величина 500 мм
- Конструкция рабочего стула должна учитывать рост пользователя, продолжительность работы; способствовать естественному движению пользователя, не оказывать дополнительной нагрузки на мышцы спины и шейно-плечевой области;
- Конструкция рабочего стола также должна учитывать естественное положение пользователя при работе за компьютером, длительность работы и обеспечивать оптимальное размещение всего используемого в процессе работы оборудования[33].

Рассмотрим более подробно основные требования к конструкции стола:

- ширина и глубина поверхности сиденья - не менее 400 мм;
- регулировка высоты поверхности сиденья в пределах 400 - 550 мм и углов наклона вперед до 15° и назад до 5°;
- высота опорной поверхности спинки 300 ± 20 мм, ширина - не менее 380 мм и радиус кривизны горизонтальной плоскости - 400 мм;
- угол наклона спинки в вертикальной плоскости – в пределах ± 30°;
- регулировка расстояния спинки от переднего края сиденья – в пределах 260 - 400 мм;

На рисунке 6.2 представлены основные требования к организации рабочего места пользователя.

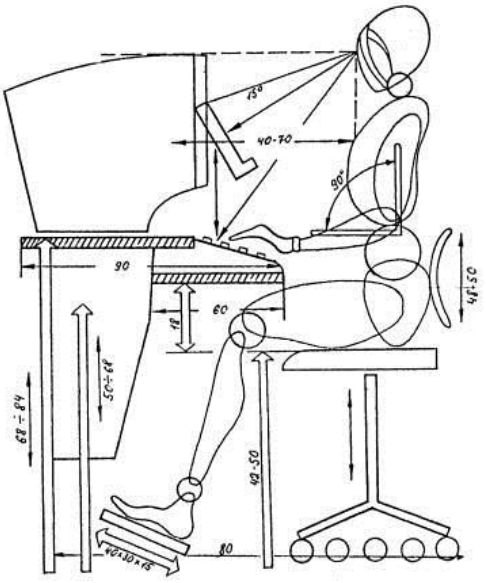


Рисунок 6.1 - Требования к организации рабочего места пользователя

6.4.2 Режим труда и отдыха при работе с компьютером

При работе с компьютером в среднем через 2 часа у пользователя наблюдается утомления. Во избежание дальнейшего ухудшения состояния пользователя и снижения его активности, необходимо соблюдать правильный режим работы и отдыха.

Трудовая деятельность на ПК делится на 3 группы в зависимости от характера выполняемой работы:

- группа А – считывание информации с экрана по запросу;
- группа Б – ввод информации;
- группа В – режим диалога с ПК, творческая работа.

При выполнении смешанного типа работ, пользователя относят к той группе, на деятельность которой он тратит не менее 50 % рабочего времени.

По степени тяжести и напряженности работы на ПК выделяют следующие группы:

- группы А и Б – суммарное число считываемой и водимой информации соответственно;
- группа В – суммарное время непосредственного диалога с компьютером[47].

В таблице 6.6 представлены категории тяжести работ в зависимости от нагрузки для каждой группы

Таблица 6.5 – Суммарное время регламентированных перерывов в зависимости от продолжительности работы, вида и категории трудовой деятельности с ПЭВМ

Категория работы с ПЭВМ	Уровень нагрузки за рабочую смену при видах работ с ПЭВМ			Суммарное время регламентированных перерывов, мин.	
	группа А, количество знаков	группа Б, количество знаков	группа В, ч	при 8-часовой смене	при 12-часовой смене
I	до 20 000	до 15 000	до 2	50	80
II	до 40 000	до 30 000	до 4	70	110
III	до 60 000	до 40 000	до 6	90	140

Для 8-часовой рабочей смены установлены следующие режимы перерывов (в зависимости от категории работы):

1) через 2 часа от начала рабочего дня, через 2 часа после обеденного перерыва – по 15 минут;
2) через 2 часа от начала рабочего дня, через 1,5 – 2 часа после обеденного перерыва – по 15 минут или через каждый час работы - по 10 минут;
3) через 1,5 – 2 часа от начала рабочего дня, через 1,5 – 2 часа после обеденного перерыва – по 20 минут или через каждый час работы – по 15 минут.

Также необходимо использовать регламентированные микроперерывы для осуществления массажа пальцев и гимнастики для глаз.

Соблюдение данных мер позволит снизить психологическую нагрузку, утомляемость, а также послужить профилактикой нарушения зрения.

Вывод

В данной главе были рассмотрены основные аспекты социальной ответственности с учетом тематики выполняемой работы. Осведомленность в данной области, а также соблюдение основных правил техники безопасности на рабочем месте позволяет организовать рабочий процесс с минимальным ущербом для его участников. Кроме того, такие вопросы, как защита окружающей среды могут послужить предметом для рационального экологического поведения пользователей персональных компьютеров.

Заключение

В диссертационной работе было проведено исследование проблемы интеграции разнородных реляционных баз данных организации с использованием технологий работы с семантикой. Разработана архитектура программной системы, включающая использование посредников-адаптеров для преобразования реляционных БД в RDF граф и основанная на MVC модели проектирования программного обеспечения. На основе использования RDF графов, был разработан алгоритм интеграции. Проведено исследование возможностей D2R, RDFLib. Создан прототип программной системы, включающей клиентскую и серверную части веб сервиса, а также платформы D2R для реализации посредников-адаптеров. Проведена апробация системы на примере реляционных СУБД: PostgreSQL и MySQL.

В результате проведенного исследования показана возможность создания эффективных систем интеграции реляционных СУБД на основе стандартных инструментов Semantic Web.

Список литературы

1. Представление и использование реляционных баз данных в виде RDF графов/ Рафиков М.Р, Тузовский А.Ф // Молодежь и современные информационные технологии: сборник трудов XIII Международной научно-практической конференции студентов, аспирантов и молодых ученых - 2015г.
2. Wiederhold, G. “Mediators in the Architecture of Future Information Systems”. IEEE Computer 25(3), 38–49. 1992
3. Bush, V. “As We May Think” The Atlantic Monthly 176(1), 101–108. 1945
4. ”Resource Integration Using a Large Knowledge Base in Carnot”/ Collet, C., Huhns, M. N. and Shen, W.-M. // Computer 24(12), 55–62. 1991
5. “InfoSleuth: agent-based semantic integration of information in open and dynamic environments”/ Bayardo, R. J., Bohrer,W., Brice, R., Cichocki, A., Fowler, J., Helal, A., Kashyap, V., Ksiezyk, T., Martin, G., Nodine, M., Rashid, M., Rusinkiewicz, M., Shea, R., Unnikrishnan, C., Unruh, and Woelk, D. // SIGMOD ’97: Proceedings of the 1997 ACM SIGMOD international conference on Management of data, ACM, New York, NY, USA, pp. 195–206. 1997
6. “Garlic: a new flavor of federated query processing for DB2”/ Josifovski, V., Schwarz, P., Haas, L. and Lin, E // SIGMOD ’02: Proceedings of the 2002 ACM SIGMOD international conference on Management of data, ACM, New York, NY, USA, pp. 524–532. 2002
7. “The TSIMMIS Project: Integration of Heterogeneous Information Sources”/ Chawathe, S. S., Garcia-Molina, H., Hammer, J., Ireland, K., Papakonstantinou, Y., Ullman, J. D. and Widom, J. // IPSJ Conference, pp. 7–18. 1994
8. “Object Exchange Across Heterogeneous Information Sources”/ Papakonstantinou, Y., Garcia-Molina, H. and Widom, J. // ICDE ’95:

Proceedings of the Eleventh International Conference on Data Engineering, IEEE Computer Society, Washington, DC, USA, pp. 251–260. 1995

9. “Scaling heterogeneous databases and the design of Disco” / Tomasic, A., Raschid, L. and Valduriez, P. // Proceedings of the 16th International Conference on Distributed Computing Systems (ICDCS 1996), Hong Kong, IEEE Computer Society, Los Alamitos, CA, USA, p. 449. 1996
10. ”Leveraging Mediator Cost Models with Heterogeneous Data Sources” / Naacke, H., Gardarin, G. and Tomasic, A. // ICDE ’98: Proceedings of the Fourteenth International Conference on Data Engineering, IEEE Computer Society, Washington, DC, USA, pp. 351–360. 1998
11. Lenat, D. B. “CYC: a large-scale investment in knowledge infrastructure” // Communications of the ACM 38(11), 33–38. 1995
12. Gong, L. “JXTA: A Network Programming Environment”// IEEE Internet Computing 5(3), 88–95. 2002
13. Andreas Langegger “A Flexible Architecture for Virtual Information Integration based on Semantic Web Concepts”// Institute for Application Oriented Knowledge Processing. 2010
14. Хабрахабр [Электронный ресурс]: Информационные интеллектуальные сети и Семантический Веб.URL: https://habrahabr.ru/post/116574 . 2011
15. Joshua Tauberer //What is RDF and what is it good for?
16. Научная библиотека Sernam [Электронный ресурс]: Работа с базами данных URL: http://sernam.ru/book_cbd.php?id=2
17. Википедия [Электронный ресурс]: Архитектура программного обеспечения. URL: https://ru.wikipedia.org/wiki/%D0%90%D1%80%D1%85%D0%B8%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F

18. Пхоун Найн/ Моделирование процессов динамического связывания Web-сервисов// диссертация кандидата технических наук : 05.13.11 Москва 2007
19. Академик [Электронный ресурс]: Django. URL: http://dic.academic.ru/dic.nsf/personal_names/3336/%D0%94%D0%B6%D0%B0%D0%BD%D0%B3%D0%BE
20. It-GOST [Электронный ресурс]: Теория и практика UML диаграмма последовательности. URL: http://www.it-gost.ru/articles/view_articles/96
21. ThisIsWhereIPostMyWork [Электронный ресурс]: Popular Internet Browsers and Operating Systems. URL: http://www.pringlessanluis.com.ar/sitio_2011/internet-explorer-9-free-download-for-mobile
22. Semantic Caching in Ontologybased Mediator Systems,/ Karnstedt, M., Sattler, K.-U., Geist, I. and Нцpfner, H.: // Berliner XML Tage, pp. 155–169. 2003
23. Зelik, T., Microformats A Pragmatic Path to the Semantic Web// Technical Report 06-01, CommerceNet Labs. 2001
24. Logical foundations of object-oriented and frame-based languages/ Kifer, M., Lausen, G. and Wu, J// ACM Journal 42(4), 741–843. 2001
25. The Information Manifold / Kirk, T., Levy, A. Y., Sagiv, Y. and Srivastava, D // Proceedings of the AAAI 1995 Spring Symp. on Information Gathering from Heterogeneous, Distributed Enviroments, pp. 85–91. 1995
26. Kooi, R.: 1980, The Optimization of Queries in Relational Databases, Phd thesis// Case Western Reserve University Cleveland, USA.
27. Kossmann, D, The State of the Art in Distributed Query Processing// ACM Computing Surveys 32(4), 422–469.2000
28. Semantic Model to Integrate Biological Resources/ Lacroix, Z., Raschid, L. and Vidal, M. E// ICDEW ’06: Proceedings of the 22nd International Conference on Data Engineering Workshops, IEEE Computer Society, Washington, DC, USA, p. 63.2006

29. Langegger, A .Semantische Datenintegration in eScience Grids// Chancen im E-Business, Invited Talk at the Austrian Federal Economic Chamber, Arbeitskreis Semantic Web, ebSemantics.
30. Sharing Data on the Grid using Ontologies and Distributed SPARQL Queries/ Langegger, A., Blochl, M. and Wob, W.://, DEXA ’07: Proceedings of the 18th International Conference on Database and Expert Systems Applications (DEXA 2007), IEEE Computer Society, Washington, DC, USA, pp. 450–454. 2007
31. Охрана труда. Основы безопасности жизнедеятельности // www.Grandars.ru. 2016. URL: http://www.grandars.ru/shkola/bezopasnost-zhiznedeyatelnosti/ohrana-truda.html (дата обращения: 22.04.2016).
32. ГОСТ 12.0.003-74. Система стандартов безопасности труда. Опасные и вредные производственные факторы. Классификация // Библиотека ГОСТов. 2016. URL: http://vsegost.com/Catalog/41/41131.shtml (дата обращения: 22.04.2016).
33. Ефремова О. С. Требования охраны труда при работе на персональных электронно-вычислительных машинах. – 2-е изд., перераб. и доп. – М. : Издательство «Альфа-Пресс», 2008. – 176 с.
34. Назаренко О. Б. Безопасность жизнедеятельности: учебное пособие / О. Б. Назаренко, Ю. А. Амелькович; Томский политехнический университет. – 3-е изд., перераб. и доп. – Томск: Изд-во Томского политехнического университета, 2013. – 178 с.
35. СанПиН 2.2.4.548-96. Санитарные правила и нормы. Гигиенические требования к микроклимату производственных помещений // Библиотека гостов и нормативов. 2016. URL: http://ohranatruda.ru/ot_biblio/normativ/data_normativ/5/5225/ (дата обращения: 23.04.2016).
36. Белов С. В. Безопасность жизнедеятельности и защита окружающей среды (техносферная безопасность): учебник / С. В. Белов. – 2-е изд., испр. и доп. – М.: Издательство Юрайт, 2011. – 680 с.

37. СанПиН 2.2.2/2.4.1340-03. Санитарно-эпидемиологические правила и нормы. Гигиенические требования к персональным электронно-вычислительным машинам и организации работы // Библиотека гостов и нормативов. 2016. URL: http://www.ohranatruda.ru/ot_biblio/normativ/data_normativ/39/39082/#i72870 (дата обращения: 23.04.2016).
38. СП 52.13330.2011. Естественное и искусственное освещение. Актуализированная редакция СНиП 23-05-95 // Докипедия. 2016. URL: http://dokipedia.ru/document/5147250 (дата обращения: 23.04.2016).
39. ГОСТ Р 12.1.019-2009 ССБТ. Электробезопасность. Общие требования и номенклатура видов защиты // Электронный фонд правовой и нормативно-технической документации. 2010. URL: http://docs.cntd.ru/document/gost-r-12-1-019-2009-ssbt (дата обращения: 24.04.2016).
40. СНиП 21-01-97. Пожарная безопасность зданий и сооружений // Библиотека гостов и нормативов. 2016. URL: http://www.ohranatruda.ru/ot_biblio/normativ/data_normativ/2/2107/ (дата обращения: 24.04.2016).
41. СанПиН 2.2.1/2.1.1.1200-03. Санитарно-эпидемиологические правила и нормативы. Санитарно-защитные зоны и санитарная классификация предприятий, сооружений и других объектов // Библиотека гостов и нормативов. 2016. URL: http://ohranatruda.ru/ot_biblio/normativ/data_normativ/11/11774/ (дата обращения: 24.04.2016).
42. СанПиН 2.1.7.1322-03. Санитарно-эпидемиологические правила и нормативы. Гигиенические требования к размещению и обезвреживанию отходов производства и потребления. 2.1.7. Почва, очистка населённых мест, бытовые и промышленные отходы, санитарная охрана почвы // Библиотека гостов и нормативов. 2016. URL:

http://ohranatruda.ru/ot_biblio/normativ/data_normativ/11/11774/ (дата обращения: 02.05.2016).

43. Постановление Правительства РФ от 03.09.2010 N 681 (ред. от 01.10.2013) "Об утверждении Правил обращения с отходами производства и потребления в части осветительных устройств, электрических ламп, ненадлежащие сбор, накопление, использование, обезвреживание, транспортирование и размещение которых может повлечь причинение вреда жизни, здоровью граждан, вреда животным, растениям и окружающей среде // Консультант Плюс. 2015. URL: http://www.consultant.ru/document/cons_doc_LAW_104420/e1b31c36ed1083efeb6cd9c63ed12f99e2ca77ed/#dst100007 (дата обращения: 02.05.2016).
44. Энергосбережение в компьютерном мире // HWP. 2008. URL: http://www.hwp.ru/articles/Energosberezhenie_v_kompyuternom_mire_CHast_1___osnovnie_tendentsii/?SHOWALL_1=1 (дата обращения: 24.04.2016).
45. НПБ 105-03 Определение категорий помещений, зданий и наружных установок по взрывопожарной и пожарной опасности // Электронный фонд правовой и нормативно-технической документации. 2016. URL: http://docs.cntd.ru/document/1200032102 (дата обращения: 24.04.2016).
46. ППБ 01–03. Правила пожарной безопасности в Российской Федерации. – М.: Министерство Российской Федерации по делам гражданской обороны, чрезвычайным ситуациям и ликвидации последствий стихийных бедствий, 2003.
47. Трудовой кодекс Российской Федерации" от 30.12.2001 N 197-ФЗ (ред. от 30.12.2015) // Консультант Плюс. 2015. URL: http://www.consultant.ru/document/cons_doc_law_34683/?utm_campaign=law_doc&utm_source=google.adwords&utm_medium=cpc&=utm_content=Labor%20Code&gclid=CjwKEAjwgPe4BRCB66GG8PO69QkSJAC4EhHhU-5yAFZCJfmzkTLNGnrpgHHAYFPhhPzRo-sZGWmqnBoCPynw_wcB (дата обращения: 25.04.2016).

Приложения