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

Институт кибернетики
Направление подготовки 09.03.02 «Информационные системы и технологии»
Кафедра вычислительной техники

БАКАЛАВРСКАЯ РАБОТА

Тема работы
Разработка программного многомодульного приложения для адаптации и верификации баз данных

УДК 004.62

Студент

Группа	ФИО	Подпись	Дата
8и2б	Поданев С.Е.		

Руководитель

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент каф. ВТ	Шерстнев В.С.	к.т.н.		

КОНСУЛЬТАНТЫ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент каф. менеджмента	Николаенко В.С.	–		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент каф. ЭБЖ	Мезенцева И.Л.	–		

По разделам «Техническое задание», «Аналитический обзор», «Проектирование приложения», «Реализация приложения»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Инженер-программист	Гриценко М.С.	–		

ДОПУСТИТЬ К ЗАЩИТЕ:

Зав. кафедрой	ФИО	Ученая степень, звание	Подпись	Дата
Вычислительной техники	Марков Н.Г.	д.т.н.		

Томск – 2016 г.

Запланированные результаты обучения по программе

Код результатов	Результат обучения (выпускник должен быть готов)
Профессиональные компетенции	
Р1	Применять базовые и специальные естественнонаучные и математические знания в области информатики и вычислительной техники, достаточные для комплексной инженерной деятельности.
Р2	Применять базовые и специальные знания в области современных информационных технологий для решения инженерных задач.
Р3	Ставить и решать задачи комплексного анализа, связанные с созданием аппаратно-программных средств информационных и автоматизированных систем, с использованием базовых и специальных знаний, современных аналитических методов и моделей.
Р4	Разрабатывать программные и аппаратные средства (системы, устройства, блоки, программы, базы данных и т. п.) в соответствии с техническим заданием и с использованием средств автоматизации проектирования.
Р5	Проводить теоретические и экспериментальные исследования, включающие поиск и изучение необходимой научно-технической информации, математическое моделирование, проведение эксперимента, анализ и интерпретация полученных данных, в области создания аппаратных и программных средств информационных и автоматизированных систем.
Р6	Внедрять, эксплуатировать и обслуживать современные программно-аппаратные комплексы, обеспечивать их высокую эффективность, соблюдать правила охраны здоровья, безопасность труда, выполнять требования по защите окружающей среды.
Универсальные компетенции	
Р7	Использовать базовые и специальные знания в области проектного менеджмента для ведения комплексной инженерной деятельности.
Р8	Осуществлять коммуникации в профессиональной среде и в обществе в целом. Владеть иностранным языком (углублённый английский язык), позволяющем работать в иноязычной среде, разрабатывать документацию, презентовать и защищать результаты комплексной инженерной деятельности.
Р9	Эффективно работать индивидуально и в качестве члена команды, состоящей из специалистов различных направлений и квалификаций.
Р10	Демонстрировать личную ответственность за результаты работы и готовность следовать профессиональной этике и нормам ведения комплексной инженерной деятельности.
Р11	Демонстрировать знания правовых, социальных, экологических и культурных аспектов комплексной инженерной деятельности, а также готовность к достижению должного уровня физической подготовленности для обеспечения полноценной социальной и профессиональной деятельности.

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

Институт кибернетики
Направление подготовки 09.03.02 «Информационные системы и технологии»
Кафедра вычислительной техники

УТВЕРЖДАЮ:
Зав. кафедрой
_____ _____ Марков И.Г.
(Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

бакалаврской работы

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

Студенту:

Группа	ФИО
8и2б	Поданев Сергей Евгеньевич

Тема работы:

Разработка программного многомодульного приложения для адаптации и верификации баз данных	
Утверждена приказом директора (номер, дата)	№ 334/с 26.01.2016 г.

Срок сдачи студентом выполненной работы:	16.06.2015

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе	Базы данных ООО «Химсофт», техническое задание на проект
Перечень подлежащих исследованию, проектированию и разработке вопросов	– выбор методологии программирования; – выбор шаблонов проектирования; – выбор соглашений по оформлению кода; – выбор средств подключений к БД; – проектирование модулей; – реализация модулей; – финансовый менеджмент; – ресурсоэффективность и ресурсосбережение; – социальная ответственность.
Перечень графического материала	– схема UML диаграммы классов приложения; – схема диаграммы потоков данных приложения в нотации Йордана/Де Марко

Раздел	Консультант

«Аналитический обзор», «Проектирование приложения», «Реализация приложения»	Гриценко М.С.
«Социальная ответственность»	Мезенцева И.Л.
«Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»	Николаенко В.С.

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	15.02.2016

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент кафедры ВТ	Шерстнев В.С.	к.т.н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8и2б	Поданев Сергей Евгеньевич		

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

Институт кибернетики
Направление подготовки 09.03.02 «Информационные системы и технологии»
Уровень образования – бакалавриат
Кафедра вычислительной техники
Период выполнения весенний семестр 2016 учебного года

Форма представления работы:

бакалаврская работа

(бакалаврская работа, дипломный проект/работа, магистерская диссертация)

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Срок сдачи студентом выполненной работы:	16.06.2015

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
31.02.2016	Аналитический обзор	15
02.04.2016	Проектирование приложения	35
25.05.2016	Реализация приложения	25
29.05.2016	Результаты разработки	5
31.05.2016	Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	10
05.06.2016	Социальная ответственность	10

Составил преподаватель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
доцент кафедры ВТ	Шерстнев В.С.	к.т.н.		

СОГЛАСОВАНО:

Зав. кафедрой	ФИО	Ученая степень, звание	Подпись	Дата
Вычислительной техники	Марков Н.Г.	д.т.н.		

Реферат

Выпускная квалификационная работа 96 с., 22 рис., 16 табл., 48 источников, 7 прил.

Ключевые слова: автоматизация, базы данных, верификация, адаптация, поставщик данных.

Объектом исследования являются базы данных ООО «Химсофт».

Цель работы – разработать многомодульное приложения для адаптации и верификации баз данных.

В процессе исследования проводились анализ выбранных СУБД, сопоставимость их моделей данных, языка запросов SQL, языков программирования, среды разработки, методологии разработки.

В результате исследования было разработано многомодульное приложение с готовым модулем конвертации в MS Access.

Степень внедрения: разработанное приложение является важным инструментом для обслуживания баз данных ЛИС «Химик-Аналитик» и является автономным приложением.

Область применения: разработанное приложение представляет интерес для организации ООО «Химсофт» и их заказчиков.

Экономическая эффективность разработки модуля составила 94258 руб. 34 коп. Уровень научного эффекта – средний.

В будущем планируется:

- доработка модуля создания SQL модификаций;
- доработка модуля проверки соответствия структуры;

Определения и обозначения

СУБД (Система управления базами данных) – совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

БД (База данных) – представленная в объективной форме совокупность самостоятельных материалов (статей, расчётов, нормативных актов, судебных решений и иных подобных материалов), систематизированных таким образом, чтобы эти материалы могли быть найдены и обработаны с помощью электронной вычислительной машины.

Верификация – подтверждение соответствия конечного продукта предопределённым эталонным требованиям.

Дамп – снимок информации о состоянии компьютерной системы.

DLL (Dynamic Link Library) — динамическая библиотека, позволяющая многократное использование различными программными приложениями.

DML (Data Manipulation Language) – это семейство компьютерных языков, используемых в компьютерных программах или пользователями баз данных для получения, вставки, удаления или изменения данных в базах данных.

WPF (Windows Presentation Foundation) – система для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем, графическая (презентационная) подсистема в составе .NET Framework.

MVVM (Model-View-ViewModel) – шаблон проектирования архитектуры приложения.

Паттерн – схема-образ, действующая как посредствующее представление.

XML (eXtensible Markup Language) – расширяемый язык разметки.

TFS (Team Foundation Server) – продукт корпорации Microsoft, представляющий собой комплексное решение, объединяющее в себе систему

управления версиями, сбор данных, построение отчётов, отслеживание статусов и изменений по проекту и предназначенное для совместной работы над проектами по разработке программного обеспечения.

XAML (eXtensible Application Markup Language) – расширяемый язык разметки для приложений, разработанный Microsoft.

Оглавление

Введение ... 9
Обзор литературы ... 11
1 Аналитический обзор ... 12
2 Проектирование приложения ... 22
2.1 Организация проекта ... 22
2.2 Выбор шаблонов проектирования ... 23
2.3 Проектирование общей архитектуры приложения ... 24
2.4 Алгоритмы приложения ... 27
2.5 Алгоритм модуля DBManagement ... 27
2.6 Алгоритм модуля ValidationModule ... 28
2.7 Алгоритм модуля PatchModule ... 29
2.8 Проектирование алгоритма модуля ConverterModule ... 30
3 Реализация приложения ... 35
3.1 Реализация модуля DBManagement ... 35
3.2 Реализация модуля DevelopmentLib ... 37
3.3 Реализация интерфейса для модулей ... 39
3.4 Реализация модуля ValidationModule ... 39
3.5 Реализация модуля PatchModule ... 41
3.6 Реализация модуля ConverterModule ... 43
4 Результаты разработки ... 46
5 Финансовый менеджмент, ресурсоэффективность и ресурсосбережение ... 48
6 Социальная ответственность ... 63
Заключение ... 77
Список использованных источников ... 78
Приложение А ... 84
Техническое задание ... 84
Приложение Б ... 88
Приложение В ... 89
Приложение Г ... 90
Приложение Д ... 91
Руководство пользователя ... 93

Введение

Существует проблема поддержки множества баз данных неквалифицированными пользователями. В текущее время прослеживается тенденция к увлечению количества обслуживаемых баз данных разных разработчиков, причём сложность СУБД этих баз данных повышается для обеспечения больших возможностей. Вместе со сложностью растут и вычислительные потребности, которые не всегда оправданы в рамках определенных задач. Поэтому для обеспечения некоторых СУБД необходимы более мощные вычислительные устройства, которые являются более дорогими. Это приводит также к экономической проблеме. Неавтоматизированное решение данной проблемы занимает существенное время программистов, так как необходимо вручную выполнить необходимые манипуляции с данными в соответствии с бизнес-логикой компании. Для уменьшения этого времени было предложено автоматизировать эту деятельность в рамках предприятия ООО «Химсофт».

Целью работы выпускной квалификационной работы является разработка многомодульного приложения для адаптации и верификации баз данных. **Объектом исследования** являются базы данных ООО «Химсофт». **Предметом исследования** является процесс переноса данных с СУБД MS SQL Server и Oracle на MS Access, так как MS Access менее требовательна к ресурсам, а также в приложение закладывается возможность по введению других функций по поддержке баз данных.

Задачами работы являются:

- анализ предметной области;
- анализ средств выполнения проекта;
- проектирование приложения;
- реализация приложения;
- анализ вопросов социальной ответственности;

- оценка с точки зрения финансового менеджмента.

В результате реализации приложения оно вводится в работу на предприятии ООО «Химсофт».

Обзор литературы

Решаемые проблемы обслуживания баз данных, миграции данных являются обсуждаемыми и имеются некоторые решения. Проблемы разных типов данных, интерфейсов и даже моделей данных решаются в решении проекта IBM и Swiss bank, где конвертируется иерархическая база данных в реляционную [1]. В статье по разработки одного из уровней программного обеспечения планирования ресурсов предприятия Fenix решается проблема миграции данных, так как пользователь использует чаще всего лишь один тип СУБД [2]. Обслуживание запросов для нескольких баз данных решается в [3], где создаётся адаптивная система. Похожая проблема разбирается в статье по моделированию онтологических семантических ограничений в реляционных базах данных [4]. В этом проекте создаются таблицы, триггеры и другие объекты в соответствие с семантическими моделями в реляционных базах данных. Таким образом предлагаются несколько решений проблемы, которая постоянно развивается в зависимости от условий проекта.

1 Аналитический обзор

На сегодняшний день происходит острое развитие компьютерных технологий как в аппаратной, так и в программной сфере. Постоянно модернизуются алгоритмы и способы обработки, хранения информации. При увеличении алгоритмов растет количество реализаций в виде различных программных продуктов. Данная тенденция прослеживается и в сфере обслуживания баз данных – появляются различные типы хранилищ по модели данных в виде реляционных, иерархических, сетевых, функциональных, объектных и т.д. Однако, количество реализаций баз данных одного типа так же велико. Например, для одних из наиболее популярных реляционных СУБД существует множество реализаций: MS SQL Server, MySQL, Oracle, PostgreSQL, IBM DB2, MS Access. У каждого из перечисленных программных продуктов существуют свои преимущества и недостатки. Выбор СУБД осуществляется в зависимости от ситуации – поставленных задач и требований. Разные компании предпочитают разные СУБД. В связи с этим зачастую возникают проблемы при миграции данных одной и той же модели хранения в разные СУБД, так как каждый из разработчиков реализовывает собственные интерфейсы доступа к данным. Похожая ситуация сложилась в организации ООО «Химсофт».[5]

Организация ООО «Химсофт» занимается автоматизацией процессов количественного химического анализа. Для автоматизации этих процессов организация разработала программный продукт лабораторную информационно-управляющую систему «Химик-аналитик». [6] Одной из частей данного программного продукта является база данных, которая хранит необходимые данные для работы системы. Для разных заказчиков данная система может устанавливаться на 3 СУБД: MS Access, Oracle, MS SQL Server. Для обслуживания различных типов баз данных необходим инструмент, который мог бы поддерживать бизнес логику системы при переносе данных из одной системы в другую, проверке и модификации данных. Перенос данных особенно

необходим для обеспечения портативности системы, так как химические анализы зачастую необходимы в дальних и труднодоступных районах, где затруднителен выход в сеть и отдаётся предпочтение компьютерам в виде ноутбука. Соответственно для такого типа устройств вычислительные ресурсы более ограничены, что вынуждает использование менее требовательных баз данных в виде MS Access с выборкой только необходимых данных. Таким образом была поставлена цель разработать многомодульное приложение для адаптации и верификации баз данных. Многие из данных функций на текущий момент выполняются вручную. Выполнение данной задачи позволит выполнять перенос данных в портативную версию БД, выполняя выборку только необходимых моделей, сократить время и упросить процесс поддержки БД различных заказчиков.

Для выполнения этой задачи необходимо первоначально выполнить анализ предметной области и сделать выбор используемых технологий для разработки для обеспечения наилучшего проектирования и реализации проекта. Анализ включает в себя анализ объекта исследования – используемые СУБД предприятия, методологию программирования, соглашения по оформлению кода, средства по управлению проектом, технические инструменты и средства по решению задачи.

Для формализации поставленной задачи было выполнено техническое задание (приложение А).

1.1 Обзор используемых СУБД

Объектами исследования являются базы данных под управлением 3 СУБД: MS Access, Oracle, MS SQL Server.

MS Access разрабатывалась как часть MS Office. Данная база данных имеет закрытый код и используется по лицензии MS Office. Так как эта реляционная база данных используется как часть офисного ПО, то системные требования данной СУБД относительно не велики. Поэтому данная СУБД

поддерживает меньший функционал по сравнению с Oracle и MS SQL Server, например, явно в Access триггеры не создаются, а поддерживаются косвенно при помощи Visual Basic for Application. Помимо было проанализировано, что на MS Access накладываются такие важные ограничения как максимальный размер файла – 2 Гб, количество связей на одну таблицу – 32. В связи с относительно меньшими системными требованиями данной СУБД, требуется выполнить конвертацию данных в именно эту СУБД, учитывая ограничения и обрезая необходимые данные. [7]

MS SQL Server – более масштабная база данных по сравнению со смежным продуктом данной компании – MS Access. В отличии от Access, MS SQL поддерживает большинство функционала современных реляционных СУБД. В зависимости от версии MS SQL Server, максимальный размер файла может составлять до 524 Пб. Преимуществом данной реляционной СУБД является интуитивно понятный интерфейс и простота обслуживания. Однако по сравнению с Oracle является менее способной к масштабированию и является моноплатформенной. [8]

Oracle является одним из самых продолжительный проектов по обслуживанию баз данных, была создана в 1977 году. Данная СУБД характеризуется повышенными системными требованиями для работы сборщика мусорных данных, но обеспечивает наиболее высокое масштабирование данных. Помимо этого, является кроссплатформенной, отлично поддерживает обработку мультимедиа-данных в отличие от MS SQL Server. Недостатками этой СУБД является сложность в настройке, сложность в подключении внешними программными продуктами, долгое время загрузки. [9]

Данные СУБД объединяет тип используемой модели данных – реляционная. Это, несомненно, упрощает конвертацию данных при переносе данных из каждой системы, так как их модель в общем случае одинакова. В связи с этим в соответствии со стандартом ISO/IEC SQL на языке SQL поддерживается большинство операций.

Существует также множество различий: несогласование существующих типов данных, описание метаданных, поставщики данных. [10]

Для согласования типов данных была проанализирована таблица выборки необходимых типов данных (приложение Б). При помощи этой таблицы возможно конвертировать данные из одного типа в другой без потерь данных, но с увеличением избыточности.

Предоставляемые базы данных могут содержать более 700 таблиц в своём составе. Для представления масштаба разнообразия количества связей и таблиц был сделан снимок части схемы данных одной из исследуемых баз данных, выполненный в Access 2013 (рисунок 1.1).

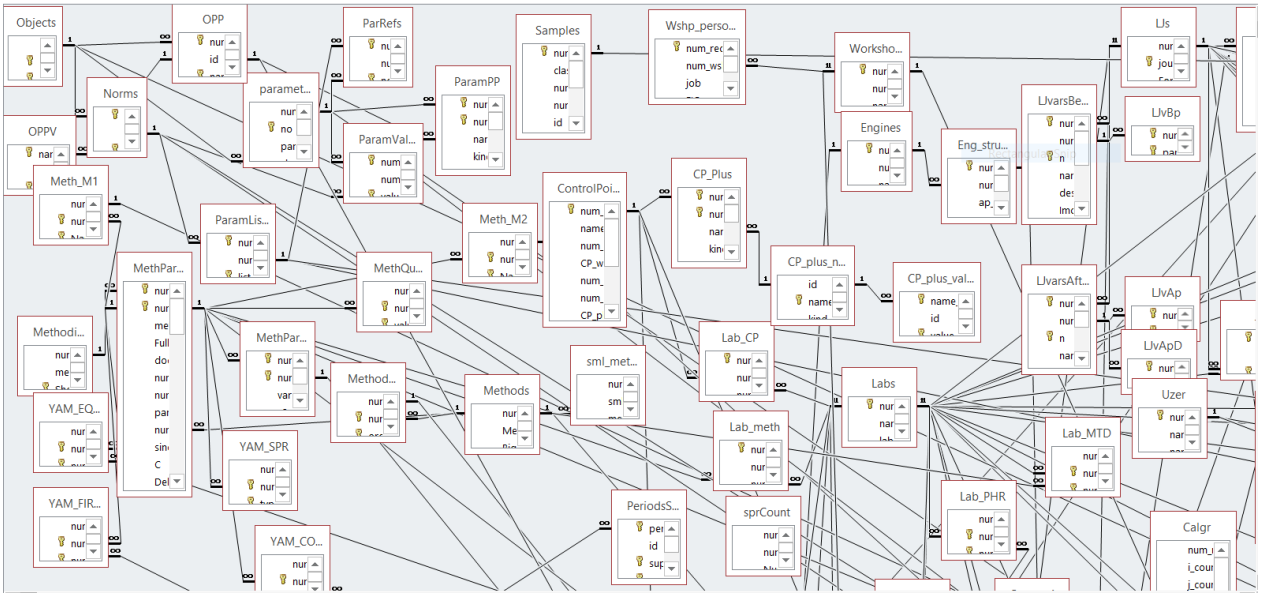


Рисунок 1.1 – Схема части обрабатываемой базы данных Access

Из этого следует вывод, что доступ к данным должен быть посредством прочтения метаданных базы данных. Использования постоянной архитектуры невозможно, т.к. построение моделей по данному количеству таблиц невозможно, а сами таблицы могут меняться.

Проанализировав типы СУБД, с которыми предстоит работать, необходимо выбрать средства для реализации программы, в том числе методологию программирования.

1.2 Выбор методологии программирования

Методология программирования — совокупность методов, применяемых на различных стадиях жизненного цикла программного обеспечения и имеющих общий философский подход. Выбор правильное методологии обеспечивать относительную простоту разработки программного продукта в зависимости от внешних и внутренних факторов. Необходимо оценить масштабность задачи, количество участников, степень ответственности, время, отведенное на разработку приложения и т.п. В соответствие с этой оценкой необходимо выбрать одну методологию, которой программист будет придерживаться до окончания проекта. Для осуществления выбора необходимо сформулировать условия выполнения проекта, для которого предстоит выбрать оптимальную методологию.

Существует огромное количество методологий, для выбора были выбраны наиболее популярные и проверенные разработчиками или наиболее близко подходящие по условиям решаемой задачи.

1.2.1 Условия выполнения проекта

Проект выполняется 1 студентом под контролем руководителя. Каждый этап контролируется и проверяется минимум 1 раз в неделю. Задача относительно просто делится на независимые блоки. Срок выполнения работы – около 5 месяцев.

1.2.2 Существующие методологии программирования

Методология программирования в виде **водопадной модели** является наиболее традиционной из всех встречающихся. Состоит из 5 блоков: определение требований, проектирование, реализация, проверка(тестирование) и поддержка. Данные блоки выполняются строго последовательно и в рамках данной модели невозможно обращение назад к предыдущему блоку.

Методология программирования в виде **спиральной модели** является другой классической моделью разработки. Данная модель подразумевает

итерационное использование раннее рассмотренной последовательности для получения лучше и лучше результата с каждой итерацией. [11]

Методология гибкого программирования **Agile** характеризуется как серия подходов к разработке программного обеспечения, ориентированных на использование интерактивной разработки, динамическое формирование требований и обеспечение их реализации в результате постоянного взаимодействия внутри самоорганизующихся рабочих групп, состоящих из специалистов различного профиля. Agile методы делают упор на непосредственное общение лицом к лицу и ставят работающий продукт приоритетнее документации.

Наиболее популярными методологиями, соответствующие Agile являются

- Scrum;
- Extreme Programming(XP).

Scrum делает акцент на разработки посредством выполнения спринтов – коротких итераций, во время которых функциональность блока определена и не изменяется. Перед тем, как приступить к новому спринту определяется наиболее важный из блоков, который реализуется на следующем спринте. Scrum позволяет модули, описанные в задании, реализовывать последовательно, в порядке важности каждого из них. В рамках методологии возможен анализ работы на любом этапе и является обязательным, что позволяет контролировать процесс работы.

XP заключается в соблюдении нескольких принципов, позволяющих ускорить работу программистов: короткий цикл разработки, парное программирование, простота, непрерывная разработка и т.п. [12]

1.2.3 Выводы по выбору методологии программирования

Методология программирования в виде водопадной модели благодаря своей простоте имеет применение во многих проектах. Однако минусы этой

модели не позволяют использовать её в рамках данного проекта: невозможность возвращения к предыдущему этапы и излишнее документирование каждого шага, чтобы получить в итоге необходимый продукт.

Методологию программирования в виде спиральной модели не является целесообразно использовать в проекте, так как данная методология содержится в других известных методологиях и при проектировании больших систем трудно переписывать результаты выполнения предыдущей итерации из-за больших объемов вычислений и очень малых задач на каждом этапе.

Парное программирование исключается в силу того, что разработчиком является один студент.

Методология Scrum является наиболее подходящей из всех методологий для выполнения проекта. Она позволяет выполнять все требования к проекту в сложившихся условиях и акцентирует своё внимание на процессе разработки конечного продукта с минимальным количеством документации, что позволяет уделять больше внимание процессу создания. Данная методология наиболее полно подходит для реализации проекта, так как процесс разработки делится на короткие итерации, результатом которых является работающий программный модуль. При этом в данной методологии постоянно возможно изменение задания на любом этапе работы, что позволяет сделать продукт более гибким и конкурентоспособным. Кроме того, проверка каждого этапа осуществляется лично руководителем, что соответствует модели.

1.3 Выбор соглашения по оформлению кода

Соглашением по оформлению кода является набор правил по написанию кода в определенном едином стиле. Соглашения используются, чтобы придать единообразие, что при усвоении правил облегчает написание кода, его редактирование и понимание другими разработчиками в команде. В результате это приводит к уменьшению времени для разработки проекта всей командой и самим автором кода. [13]

Как правило такие стандарты принимаются внутри организации и пишутся участниками проекта. Однако для многих языков программирования и архитектур программного обеспечения существуют общеизвестные стандарты, зарекомендовавшие себя в нескольких проектах и разработанные группами опытных разработчиков.

Одним из таких стандартов является регламент написания кода команды RSDN для C sharp, который был принят для написания данного проекта.

Russian Software Developer Network (RSDN) — сообщество русскоговорящих разработчиков программного обеспечения. Это сообщество было создано в 2001 году и отличает его наличие огромного количества материалов по программированию для Windows. Для разработки программного обеспечения данный регламент является знакомым для большинства разработчиков программного кода из русскоговорящих стран. Он содержит важные правила оформления таких структур кода как интерфейсы, атрибуты, перечисления, свойства, параметры, инструкции и т.д. [14]

Стандарт программирования в большинстве случаев ограничивается лишь стилем написания кода без ограничений на семантику кода. Для улучшения понимания кода была выбрана для изучения и применения принципов книга «Читаемый код или программирование как искусство». Эта книга учит основным принципам написания кода от правил именования переменных, заканчивая правилами организации кода. [15]

1.4 Средство управления проекта

Существует множество средств управления проектом, в том числе систем: Bitrix24, Trac, Asana, Apollo и т.д. [16].

Из всех этих системы была выбрана система Team Soundation Server (TFS) Visual Studio Online. Помимо управления проекта TFS предоставляет портал проекта, где возможно отслеживать вносимый код, формировать отчетность, сроки, статусы, вносить записи. Этот программный продукт также удобен тем,

что содержит широкую поддержку проектов в рамках языка программирования C#, который является требованием предприятия, и может иметь интеграцию в проект посредством среды разработки Visual Studio, что являлось ключевым аргументом для выбора этой системы. [17]

1.5 Средство создания интерфейса

Одним из требований предприятия является создание приложения на языке программирования C# (C sharp). [18]

Существует относительно небольшой выбор в средствах создания интерфейса для настольных приложений C#, самые используемые из них: WinForms, WPF, Microsoft Silverlight [19].

Для создания интерфейса системы необходимо использовать Windows Presentation Foundation. Microsoft Siverlight является устаревшей технологией, которая сегодня не поддерживается, поэтому её использование не является рациональным. WinForms не предлагает такой богатый выбор возможностей, предлагая в основном в своем составе статичные окна в стиле Windows. WPF — система для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем, графическая подсистема в составе .NET Framework, использующая язык XAML. Гибкость настройки интерфейса послужило решением для принятия данного средства создания интерфейса. Несмотря на то, что на усвоение языка XAML может потратится дополнительное время, эта система позволяет построить визуально привлекательный шаблон интерфейса окна приложения, тем самым позволяя сделать построение интерфейса динамичным.

1.6 Технология доступа к данным

Для языка программирования C# существуют 2 технологии доступа к данным: ADO и ADO.NET [20].

В качестве технологии доступа данных необходимо использование технологии ADO.NET. Эта технология описывается как предоставляющая

доступ к данным для приложений, основанных на Microsoft .NET, позволяет соединятся с базой данных и выполнять необходимые операции с большей поддержкой (передача данных в виде XML, асинхронные сообщения), нежели технология ADO.

Для ADO.NET доступно множество поставщиков данных. Однако, 2 из 3 используемых СУБД являются продуктами компании Microsoft, поэтому целесообразно использовать поставщики данных .Net: ODBC, OLE DB, SQL Server, Oracle, EntityClient, SQL Server Compact. [21] [22]

Среди перечисленных поставщиков поддержку нескольких баз данных осуществляют ODBC, OLE DB, EntityClient, однако так как структура базы данных заранее не известна, то технология EntityClient не может быть использована в силу невозможности работы по моделям, которые не имеется возможным заранее построить. OLE DB является более продвинутой разработкой (1996) по сравнению с ODBC (1992) и имеет более широкую поддержку баз данных, поэтому в качестве поставщика данных для Access и SQL Server был выбран OLE DB. Для упрощения развертывания приложения для взаимодействия с СУБД Oracle был выбран поставщик данных ODP(Oracle Data Provider) для платформы .NET, так как для поддержки OLE DB требуется дополнительная установка клиентского приложения. [23]

2 Проектирование приложения

Проектирование приложения заключается в работе по планированию структуры приложения и состоит из этапов:

- организация проекта;
- выбор шаблонов проектирования;
- определение общей архитектуры
- определение алгоритмов работы модулей приложения.

Организация проекта обеспечивает начальными средствами для управления и хранения результатов проекта.

Шаблоны проектирования позволяют определиться с шаблонами типичных структур проекта. Данные структуры представляют собой наиболее высокую степень абстракции построения проекта и позволяют перейти затем к выполнению проектирования приложения на более детальном уровне – на уровне классов.

Определение общей архитектуры приложения позволяет ознакомиться со взаимосвязью модулей и их ролью в приложении.

Алгоритмы работы приложения позволяют ознакомиться с независимыми от языка программирования и структуры реализации порядком выполнения некоторых из функций.

После выполнения проектирования возможен переход к реализации приложения, которое рассматривает детальное выполнение требований, заложенных в проектировании.

2.1 Организация проекта

Для создания проекта используется TFS Visual Studio Online, которое содержит в себе репозиторий и средства для управления командой. Был создан проект по адресу https://msgritsenko.visualstudio.com/DefaultCollection/SMDB. В этом проекте каждую неделю создавалась новая итерация в соответствие с методологией разработки Agile с новыми задачами.

Также в данной среде размещается код приложения, позволяющий сохранять версии программного продукта, позволяя просматривать и производить откат предыдущих версий.

Задачи в проекте делятся на 2 вида: задачи по устранению ошибок в программном коде и задачи по написанию компонентов продукта. Динамика выполнения задач отображается в виде графика работ.

2.2 Выбор шаблонов проектирования

Шаблоны проектирования необходимы для облегчения понимания логики программы, её поддержки и поддержания структурированности при масштабировании. Для каждой задачи существует свой шаблон проектирования и не существует универсального шаблона проектирования. В связи с этим необходимо выбрать шаблоны, наиболее подходящие для создания программного приложения. Для выбора шаблонов был использован источник [24], который содержит описание множества шаблонов проектирования.

Для проектирования модулей необходимо использовать **шаблон функционального дизайна**, так как гарантирует минимальное воздействие модулей на другие модули.

Фабричный метод может обеспечить многомодульность приложения за счёт создания единого интерфейса модуля приложения и создания различных реализаций данного интерфейса.

Прототип может активно использоваться при конвертации моделей данных из одной базы данных в другую, так как предпочтительнее создание новой модели, используя модель другой системы, нежели редактировать модель изначальной базы данных. Иначе при использовании одного и того же объекта возможно смешать логику исходной базы данных и конвертированной.

Одиночка может использоваться для создания единственных экземпляров данных в потоке, например, конвертора. Так как создания нескольких конверторов может обеспечить сбой при доступе к данным.

Фасад может быть применён при выводе всего функционала модуля для других программ.

Цепочка обязанностей является популярным шаблоном, так как позволяет распределить обязанности от корневой вызываемой функции общего функционала к функциям более специализированным, осуществляя всю цепочку вызова методов путём вызова простых функций, тем самым упрощая понимания каждого метода.

Наблюдатель может использоваться при разработке интерфейса приложения. WPF задает архитектурный паттерн MVVM. Посредством добавления наблюдателя возможно задание изменения данных, представленных в интерфейсе, при изменении модели данных.

Посетитель необходим при конвертации базы данных, так как модели, создаваемые в одном модуле, преобразуются в другие посредством множества внешних компонентов.

Таким образом для проектирования архитектуры приложения были выбраны шаблоны проектирования: функциональный дизайн, фабрика, прототип, одиночка, фасад, цепочка обязанностей, наблюдатель, посетитель. Приведенные доводы по воплощению тех или иных шаблонов в различных частях программы доказывают необходимость применения данных шаблонов проектирования. Благодаря этим шаблонам поддерживается структурированность и осуществляется облегчение поддержки приложения для дальнейшей разработки и масштабирования.

2.3 Проектирование общей архитектуры приложения

Так как проектируемая система предполагает создание многомодульного приложения, то необходимо создать архитектуру системы, которую возможно масштабировать за счёт подключаемых модулей и отвечала бы требованиям функционального дизайна.

Система состоит из наборов связанных компонентов, представленных на рисунке 2.1.

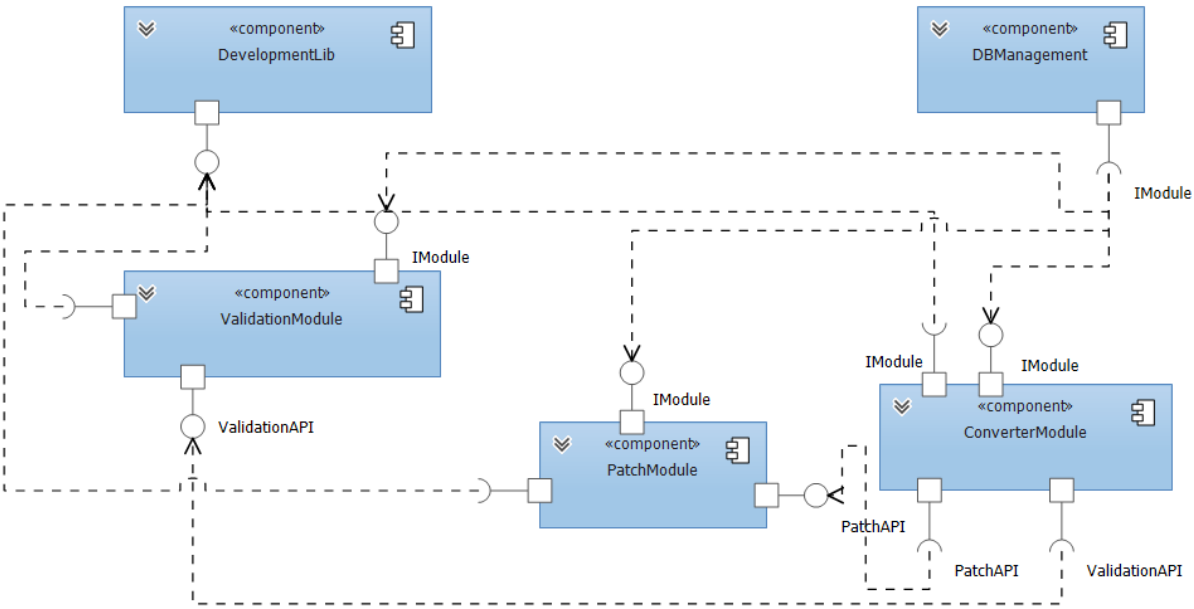


Рисунок 2.1 – Диаграмма компонетов UML приложения

Данная диаграмма является развитием схемы, представленной в техническом задании. В дополнение к ранее разработанной схеме вводится библиотека DevelopmentLib.

Компонент **DBManagement** является стартовым компонентом при запуске приложения и представляет интерфейс для выбора одного из представляемых модулей для выбранных баз данных. Т.е. он отвечает за предоставление списка баз данных и загрузку функционала модулей. Список баз данных загружается из заранее сохраненного файла, в котором указана метаинформация о базе данных, в том числе тип базы данных и её расположение.

Компонент **DevelopmentLib** является общей библиотекой создания каждого из модулей. В ней содержатся общие классы, задающие модели данных системы. В этой библиотеке созданы и полностью описаны модели соединений баз данных и интерфейса для подключения модулей, описаны объекты базы данных такие как таблица, колонка, ключи, индексы, представления. Для разрешения преобразования типов данных из одной базы данных в другую в нём

также располагается метод-загрузчик типов данных, сопоставимых в каждой из 3 баз данных.

ValidationModule – модуль валидации баз данных. Данный модуль выполняет загрузку структуры выбранной базы данных. Структура представляется в виде моделей, описанных в DevelopmentLib.

PatchingModule – модуль генерации патча SQL. Данный модуль генерирует SQL для выбранной модели данных, чтобы затем этот код SQL возможно было бы использовать.

ConvertorModule – модуль конвертации БД. Данный модуль получает на вход строку подключения к базе данных и выполняет процесс создания новой базы данных MS Access и совершает перенос структуры данных и данных из указанной базы данных, выбирая лишь необходимые данные.

Для поддержки и запуска модулей необходимо сначала спроектировать и разработать стартовый компонент DBManagement.

2.4 Алгоритмы приложения

По выбранной архитектуре была спроектирована блок схема работы приложения, представленная в приложении В. Алгоритм приложения заключается в последовательном преобразовании данных посредством 3 модулей и последовательном заполнении новой базы данных. Первоначально производится оценка количества данных для оценки количества действий по переносу данных. Затем сначала переносится структура данных в виде таблиц, заполняются эти таблицы данными, накладываются ограничения. Такая последовательность была выбрана, так как MS Access позволяет накладывать ограничения на заполненные таблицы. Если изменить порядок, вставляя сначала ограничения, то время на обработку вставляемых данных увеличится, станет возможным изменение первоначальных данных СУБД, к которой применили ограничения, что нежелательно.

Для рассмотрения используемых входных и выходных данных на каждом этапе работы приложения возможно рассмотреть этот алгоритм на диаграмме потоков данных приложения Г.

Для детального рассмотрения работы приложения рассмотрим алгоритм работы каждого модуля в отдельности.

2.5 Алгоритм модуля DBManagement

В первую очередь модуль DBManagement должен предоставлять интерфейс для загрузки модулей приложения. Для этого при запуске приложения он должен осуществлять поиск файлов модулей, представленных библиотеками .dll, как средство поддержки интеграции в приложениях технологии ASP.NET. Для построения интерфейса должна строиться модель для перечисления метаданных о загружаемом модуле, в том числе изображения, на основании которой предоставляется информация пользователю.

После выбора запускаемого модуля формируется список соединений с базами данных, с которыми осуществляется взаимодействие в рамках

исполнения выбранного модуля. Затем выбранные соединения передаются запускаемому модулю.

Для обобщения перечисленной логики компонента был составлен алгоритм работы компонента (рисунок 2.2).

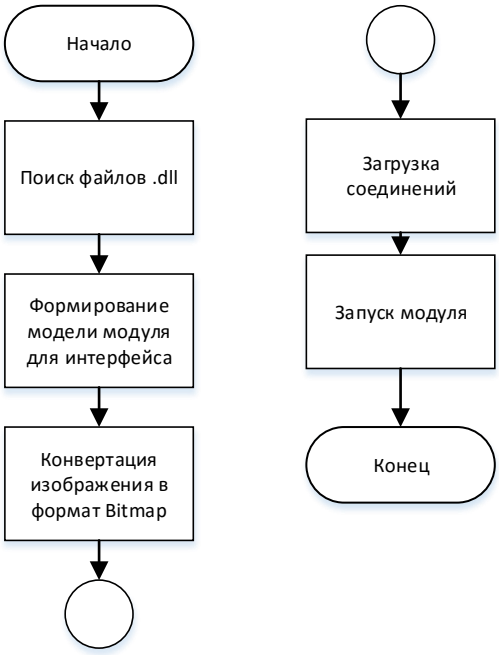


Рисунок 2.2 – Блок схема работы компонента DBManagement

Из приведенной блок схемы можно сделать вывод, что приведенный алгоритм является линейным и сложность представляет реализация перечисленных функций.

2.6 Алгоритм модуля ValidationModule

ValidationModule осуществляет загрузку структуры базы данных для обеспечения конвертации базы данных.

Для выполнения данной задачи модуль сначала определяет необходимые данные, которые запрашивает его другой модуль. В соответствии с шаблоном проектирования фабрика для выбранной задачи назначаются специальные средства для обработки запроса. Эти средства прежде всего выполняют проверку исходных данных – соединения с базой данных, формируют SQL запрос, поддерживаемый выбранным поставщиком данных и выполняют его. На основании полученных данных производиться формализация в модели,

определенными в приложении для всех модулей, которая передаётся как результат выполнения вызывающему модулю.

Для обобщения алгоритма работы модуля была составлена блок схема, представленная на рисунке 2.3.

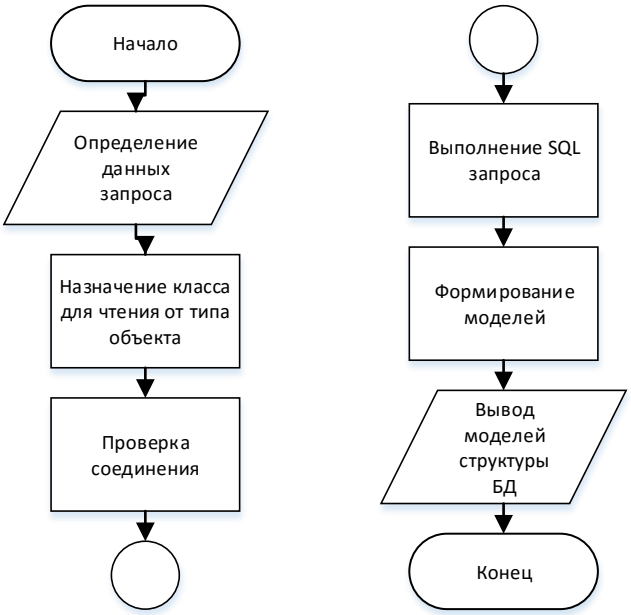


Рисунок 2.3 – Блок схема работы модуля ValidationModule

Данный алгоритм, указанный в блок схеме, демонстрирует реализацию шаблона проектирования цепочка обязанностей, что позволяет разбить задачу на элементарные и решать в зависимости от входных параметров.

2.7 Алгоритм модуля PatchModule

PatchModule выполняет создание SQL кода. Для создания SQL кода необходимо сначала получить исходные данные: тип используемой СУБД, модели данных, для которых необходимо создать SQL код, тип операции (создание, удаление, добавление и т.п.). Согласно полученным исходным данным назначаются инструменты для обработки полученного запроса, которые создают необходимый код и передают его как результат выполнения. Для обобщения перечисленного алгоритма была составлена блок схема на рисунке 2.4.

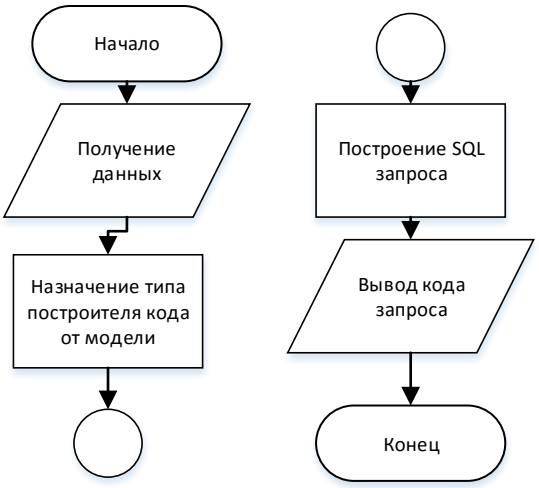


Рисунок 2.4 – Блок схема алгоритма модуля PatchModule

Данная блок схема демонстрирует одинаковый стиль проектирования, как и в модуле ValidationModule, что позволяет улучшить поддержку приложения.

2.8 Проектирование алгоритма модуля ConverterModule

Модуль ConverterModule осуществляет конвертацию баз данных из Oracle и MS SQL Server в MS Access. Конвертация базы данных включает в себя несколько этапов. Структура данного модуля и алгоритмы его выполнения являются более сложными по сравнению с ранее рассмотренными модулями и осуществляет взаимодействие с другими модулями приложения: ValidationModule и PatchModule. Для рассмотрения взаимодействия и жизненного цикла необходимо рассмотреть диаграмму последовательности (рисунок 2.5).

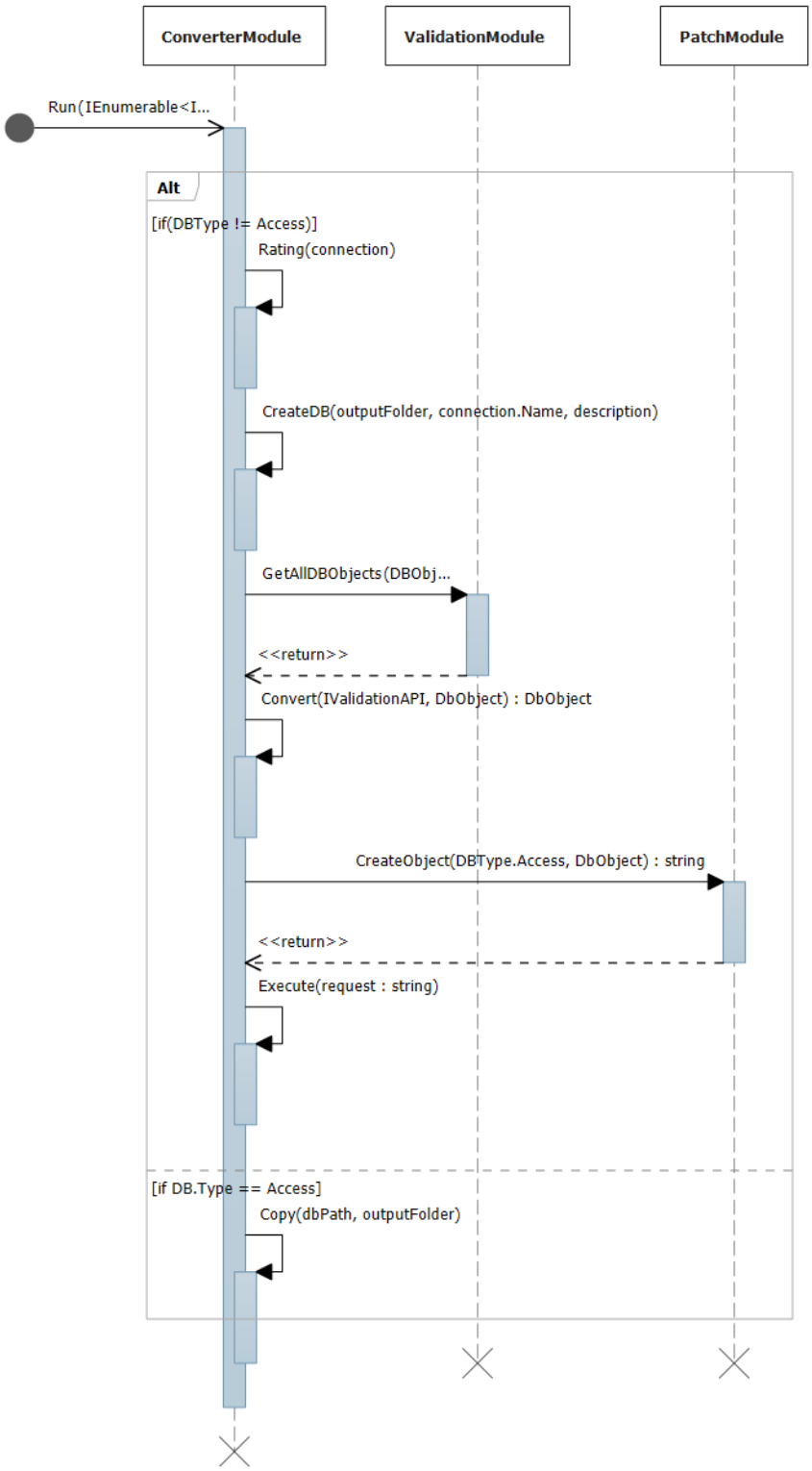


Рисунок 2.5 – Диаграмма последовательности для выполнения ConverterModule

Если исходная база данных уже является базой данных типа Access, то конвертацию базы данных проводить не нужно, достаточно лишь скопировать её в папку с результатом конвертации. Иначе база данных проходит несколько

этапов конвертации: создаётся новая база данных Access, оценивается количество действий для конвертирования, из ValidationModule извлекается структура базы данных, структура конвертируется в ConverterModule, по этой структуре вырабатывается код создания объекта в MS Access, команда вставки данных; затем полученный результат выполняется на новом сервере из ConverterModule.

Копирование базы данных производится порциями данных, равных размеру кластера на диске, что обеспечивает оптимальную скорость копирования. Для описания алгоритма копирования была использована блок-схема, представленная на рисунке 2.6.

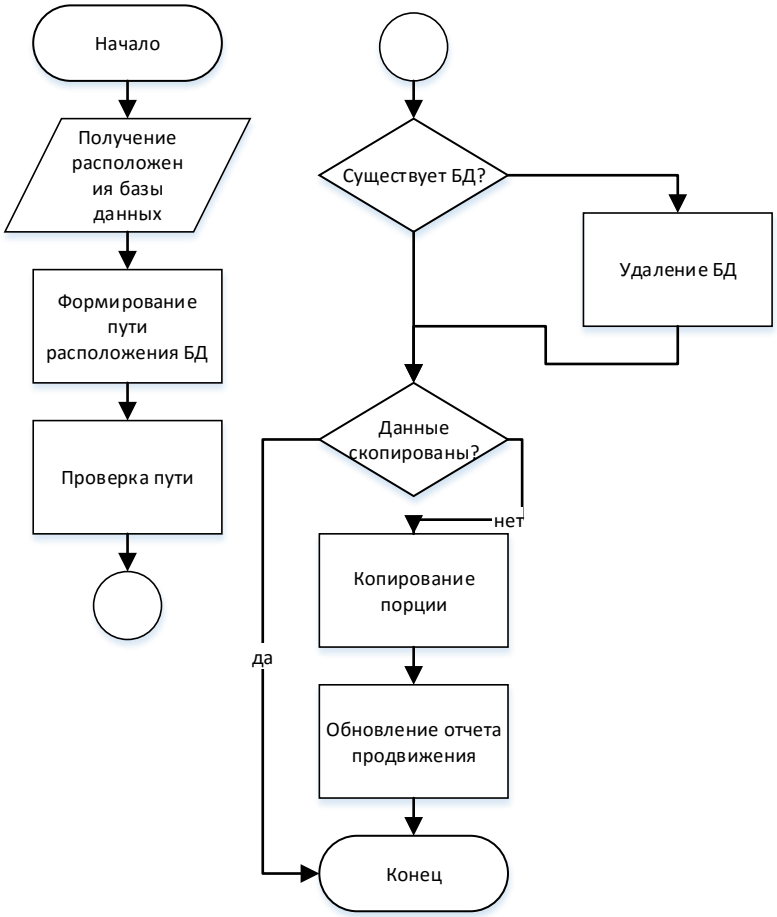


Рисунок 2.6 – Алгоритм копирования базы данных MS Access

Если выбранная база данных не управляется СУБД MS Access, то первоначально производится оценка базы данных, чтобы предоставить пользователю ход выполнения дальнейшей конвертации.

Оценка производится путем получения количества структурных элементов базы данных посредством ValidationModule, получением количества строк из всех таблиц базы данных, фильтрации повторяющихся таблиц, которые могут встречаться в Oracle из-за чувствительности к регистру. Алгоритм работы оценки отображен на рисунке 2.7.

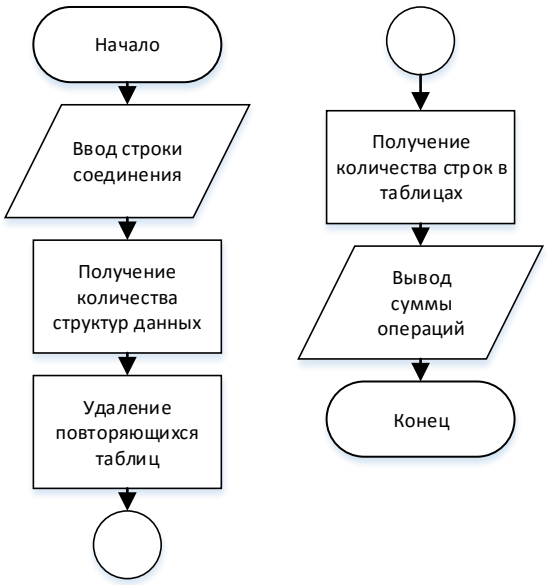


Рисунок 2.7 – Блок-схема алгоритма оценки

Конвертация данных производится путем загрузки таблицы данных, представленных в приложении А. По этой таблице соотносятся типы данных колонок и формируются новые типы данных, характерные для MS Access, перенося все параметры типов данных таких как длина строки, длина числа до и после запятой. Если тип не был найден, то необходимо формировать сообщение для пользователя. Данный алгоритм работы был представлен на рисунке 2.8.

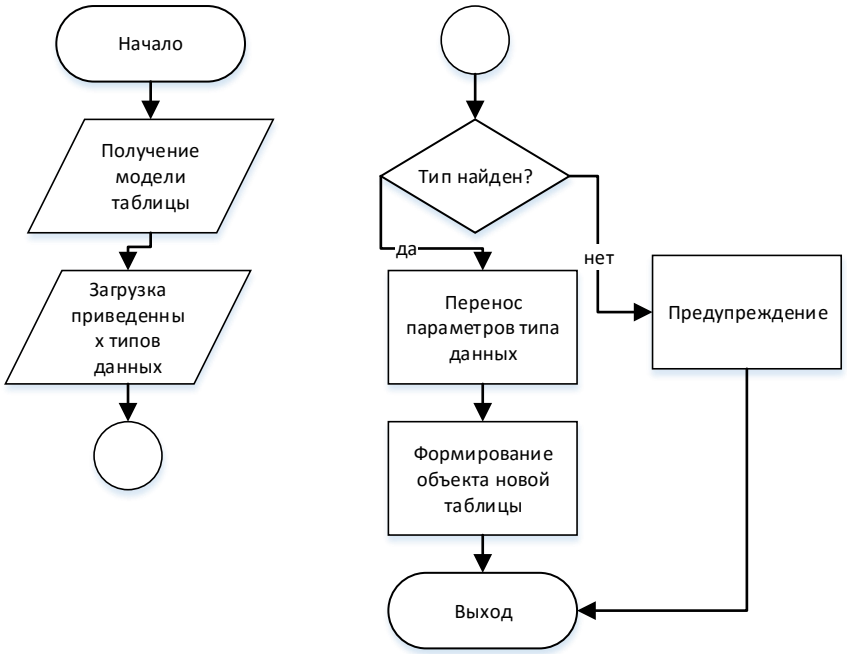


Рисунок 2.8 – Блок схема конвертации таблицы

Перенос данных из одной базы данных в другую осуществляется способом, подобным в PatchModule. Сначала формируется шаблон текста запроса, в который вставляются параметры модели. Затем запрос исполняется в базе данных. Таким образом сначала извлекаются строки данных из выбранной для конвертации базы данных, затем вставляются в новой базе данных Access.

3 Реализация приложения

Реализация приложения заключается в детальной реализации спроектированной структуры. На этом этапе архитектура программы прорабатывается на уровне классов и зависит от реализации выбранными средствами: языка программирования, средств подключения, алгоритма работы, шаблонов программирования.

Для рассмотрения реализации приложения рассмотрим разработку каждого из модулей в отдельности.

3.1 Реализация модуля DBManagement

DBManagement является стартовым окном и производит действия по загрузке модулей и данных подключения к базам данных. В соответствии с требуемыми функциями была разработана архитектура модуля в виде диаграммы классов, представленная на рисунке 3.1.

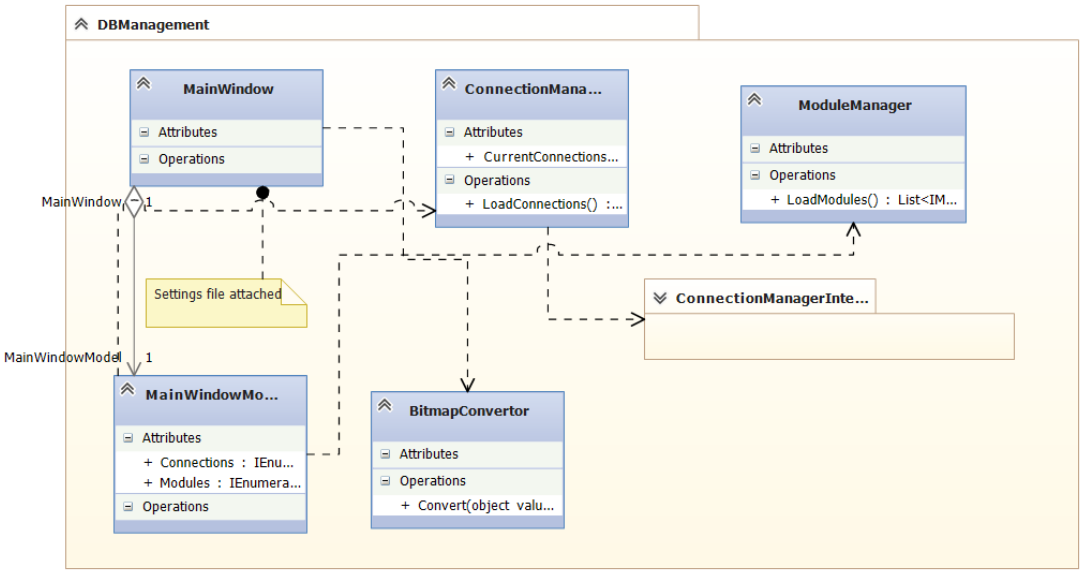


Рисунок 3.1 – Диаграмма классов UML для модуля DevelopmentLib

Являясь стартовым компонентом, представляет интерфейс для выбора модулей и строк подключения при помощи WPF окон MainWindow. Загрузка данных производится при помощи классов ConnectionManager и ModuleManager. Для загрузки информации о базах данных используется XML файл, который десериализуется и приводится к формату IConnection. Подключение модулей

происходит за счёт поиска локальных DLL файлов и стандартного механизма динамического подключения библиотек. Полученные данные сохраняются в модель главного окна MainWindowModel. При помощи паттерна MVVM и WPF происходит подстановка данных модели в шаблон представления, написанный на языке разметки XAML. [25][26][27]

В процессе реализации интерфейса главного окна и окна выбора баз данных были созданы эскизы окон, согласно которым необходимо выполнить написание шаблона XAML (рисунок 3.2, 3.3).

Рисунок 3.2 – Экскиз главного окна в виде проволочной диаграммы

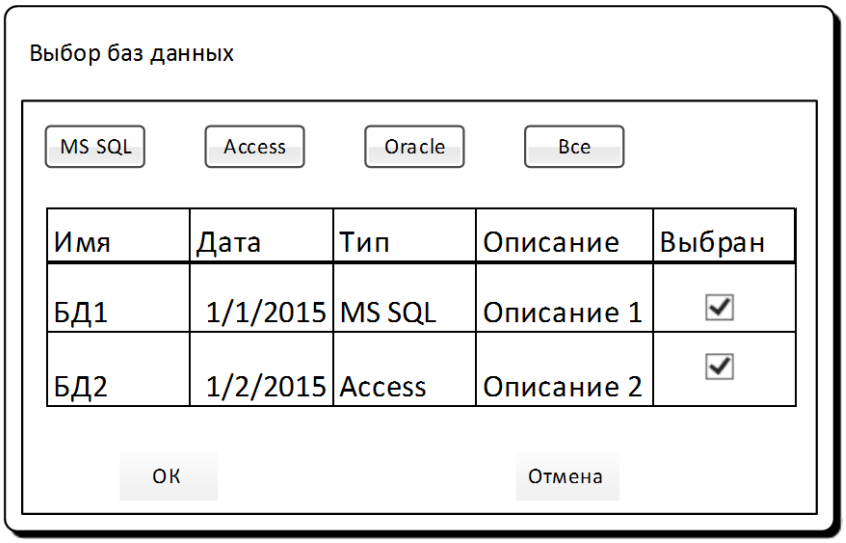


Рисунок 3.3 – Экскиз окна выбора баз данных в виде проволочной диаграммы

Для написания разметки главного окна в виде плиток, представляющих каждый из модулей возможно использовать контейнер WPF WrapPanel, который

равномерно распределяет плитки в окне. Для вывода иконки модуля необходимо создать класс-конвертер BitmapConvertor, который преобразует картинку в тип Bitmap. Для включения этого конвертера в язык XAML необходимо унаследовать его от интерфейса IValueConverter.[28][29]

Эскиз окна выбора баз данных возможно выполнить при помощи контейнера WPF GridView, представляющее таблицу данных. Кнопки в верхней части окна осуществляют фильтрацию подключений к базам данных по типу базы данных. [30]

К каждой из кнопок была прикреплена реализация интерфейса ICommand RelayCommand, которая позволяет создавать команды при помощи лямбда выражений, что значительно упрощает написание кода обработки нажатия кнопок. [31]

Согласно поставленному заданию, необходимо реализовать модуль конвертации данных с поддержкой других видов модулей. Количество модулей не является ограниченным и может расширятся. Поэтому необходимо разработать общий интерфейс доступа к модулям, который может осуществлять поддержку бизнес-логики приложения.

3.2 Реализация модуля DevelopmentLib

Данный модуль является библиотекой, предоставляющей общую логику, используемую в разных модулях приложения. Он включает в себя шаблон модулей, соединение с базой данных, объектные модели баз данных, функцию загрузки сопоставленных типов различных баз данных.

Для реализации написанного была сформирована схема классов для данной библиотеки (рисунок 3.4).

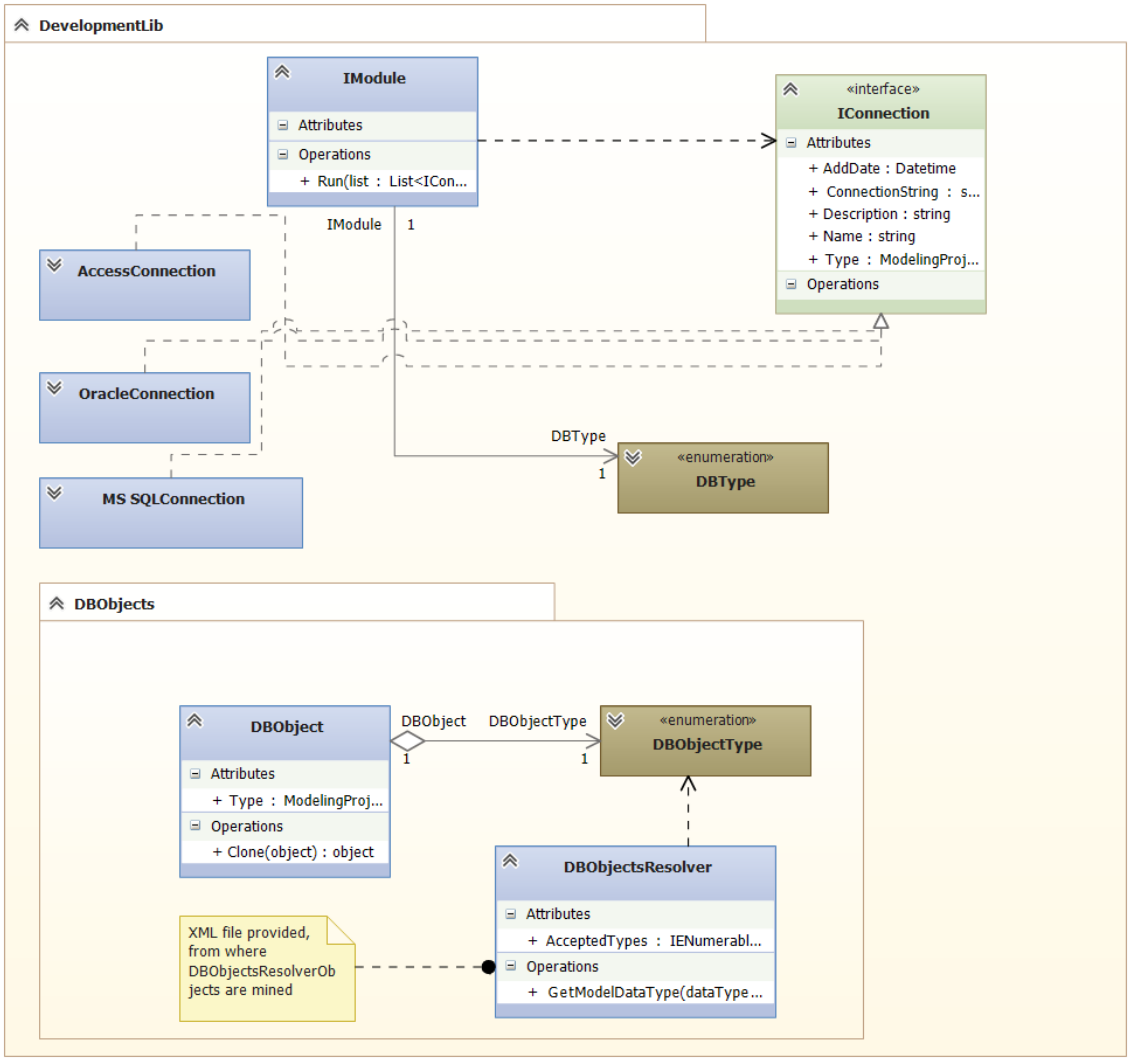


Рисунок 3.4 – Диаграмма классов UML для библиотеки DevelopmentLib

Помимо описанного шаблона модулей в данной библиотеке описывается соединения каждой из баз данных. Данные соединения объединяются при помощи интерфейса IConnection.

Класс DBObjectsResolver осуществляет загрузку сопоставленных типов баз данных из файла XML. Данный файл XML является реализацией сериализации таблицы приложения Б. Однако вместо простой подстановки типов данных также анализируются параметры типа данных. Например, строка nvarchar2(@(X)), заданная в этом файле может походить для типа данных varchar2(10) и varchar2(20) и быть конвертирована в тип данных varchar(10) и varchar(20), однако если аргументы в файле XML заданы явно и совпадают с анализируемой строкой, то данный тип данных сопоставляется в приоритете.

Формирование файла XML типов данных производится при помощи файла типовых данных ООО «Химсофт».

Перечисления служат справочниками системы по заданию типов моделей базы данных, типов поддерживаемых баз данных, обеспечивают упрощенное распознавание объектов при высоком уровне абстракции данных.

3.3 Реализация интерфейса для модулей

На рисунке 3.5 представлена разработанная схема шаблонов модулей.

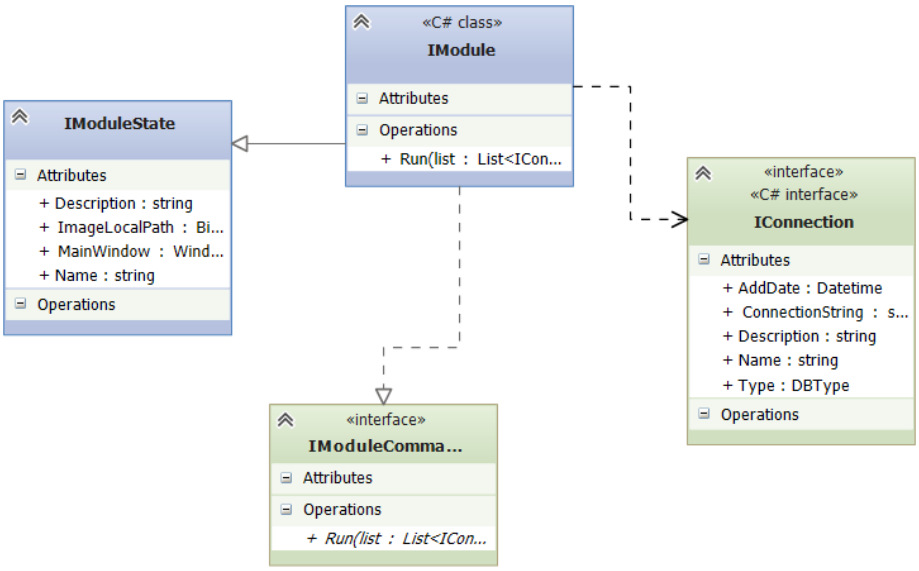


Рисунок 3.5– Диаграмма классов UML шаблона модулей

Данный шаблон был реализован при помощи шаблона проектирования команда. Центральным классом, который наследует любой из модулей является абстрактный класс IModule. Этот класс определяется единственной командой запуска, определяемой другим интерфейсом. Описание модуля хранится в классе IModuleState. В качестве параметров запуска модуля служит интерфейс описания подключения к базе данных IConnection.

После проведения реализации стартового компонента, библиотеки разработки модулей возможно перейти к реализации самих модулей.

3.4 Реализация модуля ValidationModule

Модуль ValidationModule осуществляет определение структуры баз данных, проверку структуры базы данных. Для переноса данных связей и

счетчиков используется внешняя «идеальная» база данных MS Access, которая ограничивает число связей в базе данных.

Для данного модуля была спроектирована диаграмма классов (рисунок 3.6).

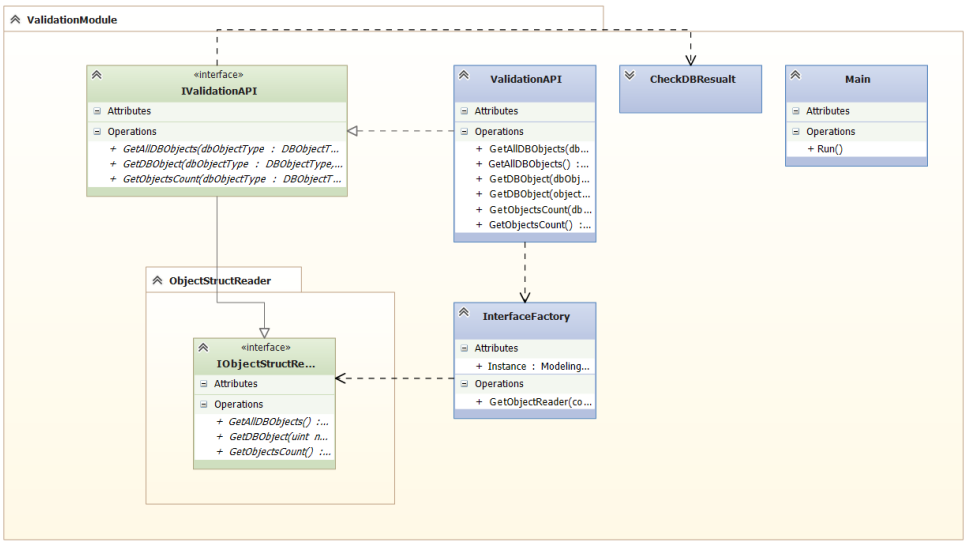


Рисунок 3.6 – Диаграмма классов UML для модуля ValidationAPI

Любой из модулей ValidationModule, ConverterModule, PatchModule должны содержать класс Main, унаследованный от интерфейса IModule, так как данные модули являются динамически подключаемыми библиотеками. Интерфейс IValidationAPI предоставляет программный интерфейс для взаимодействия других компонентов с данной библиотекой, выполненный по паттерну фасад. Синглетон InterfaceFactory является классом для выбора конкретной реализации чтения структуры данных определенного типа объекта базы данных, например, таблицы, представления. Интерфейс ObjectStructReader задаёт 3 операции по чтению типа объекта: чтение всех объектов, чтение объекта по номеру, получение количества объектов. Для каждой из СУБД в свою очередь определяется собственный класс для чтения данного типа объекта, так как запросы специфичны для каждого вида базы данных.

Чтение структуры определенного типа объектов производится при помощи выполнения SQL запроса в составе IDbCommand, созданного из

соединения IDbConnection или при помощи получения данных из таблиц метаданных, предоставляемых поставщиком данных.

Для выполнения SQL запросов, которые получают данные, необходимо сформировать коды SQL, характерные для СУБД Oracle и MS SQL Server. Данные запросы используют возможности таблиц метаданных, представляемых каждой из СУБД. Для Oracle возможно обращение к таблицам объектов all_objects, для MS SQL – таблицы схемы information_schema. Приводя результаты запросов в единый вид, таблица результатов преобразуется в коллекцию моделей, объявленных в библиотеке DevelopmentLib. Для пояснения приведем примеры запросов для получения данных таблиц. [32][33]

MS SQL Server:

```
Select COLUMN_NAME, DATA_TYPE, CHARACTER_MAXIMUM_LENGTH, TABLE_NAME,
COLUMNPROPERTY(object_id(INFORMATION_SCHEMA.COLUMNS.TABLE_NAME), COLUMN_NAME, 'IsIdentity') as isAutoColumn from INFORMATION_SCHEMA.COLUMNS Order by TABLE_NAME
```

Oracle:

```
Select TABLE_NAME from USER_TABLES
Select COLUMN_NAME "name", nullable "is_null", CONCAT(CONCAT (CONCAT (data_type,'('),data_length),')') "type"
FROM USER_TAB_COLUMNS WHERE TABLE_NAME='{0}'
```

Другие структуры извлекаются из базы данных подобным образом.

Данный модуль разрабатывается для выполнения конвертации базы данных, его графический интерфейс и связанные с ним функции планируется выполнить при развитии проекта.

3.5 Реализация модуля PatchModule

Модуль PatchModule разрабатывается для генерации SQL кода модификации данных.

Для данного модуля была спроектирована схема диаграммы классов (рисунок 3.7).

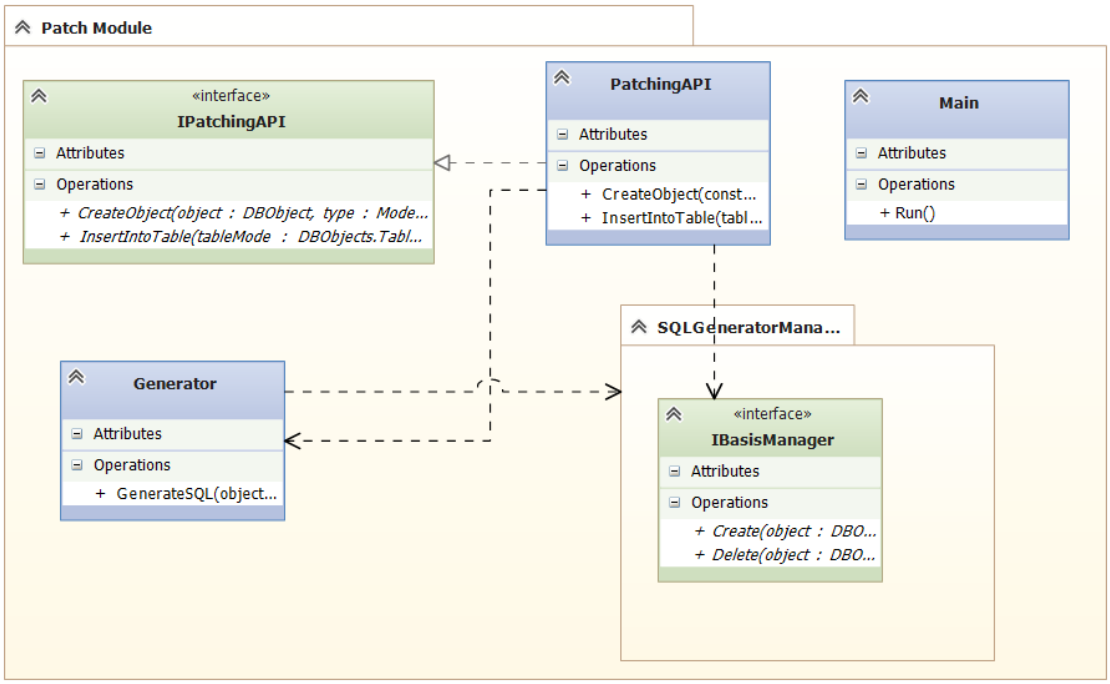


Рисунок 3.7 – Диаграмма классов UML для модуля PatchModule

Данный модуль содержит класс запуска компонента Main. Для запуска методов при помощи других модулей используется PatchingAPI, выполненный в соответствие с паттерном фасад. Для осуществления конвертации баз данных требуется генерация объектов и вставка в таблицу. SQL запрос создания объекта возвращается в виде текста. Вставка в таблицу возвращается в виде команды IDbCommand, которая содержит параметры в виде данных вставки в строку. Данное решение было принято, чтобы абстрагироваться от типа данных и вставлять данные в виде объектов, не приводя их к конкретному типу. При помощи класса Generator выбирается конкретная реализация IBasisManager в зависимости от типа создаваемых данных. При помощи способа, схожим с технологией создания динамических SQL запросов формируется код запроса, соответствующей принятой модели данных.

Для понимания процесса рассмотрим процесс формирования кода для создания таблицы.

В компоненте должен содержаться код SQL – шаблон для вставки таблицы:

```
create table [{0}]({1});
```

Затем осуществляется вставка параметров по полученной модели таблицы. Вместо “{0}” вставляется название таблицы, вместо “{1}” – строки, описывающие колонки таблицы. При этом учитывается если поле является автоинкрементным. Подобным образом происходит создание SQL кода для других структур базы данных – посредством вставки полученной модели данных в SQL шаблон кода.

Данный модуль разрабатывается для выполнения конвертации базы данных, его графический интерфейс и связанные с ним функции планируется выполнить при развитии проекта.

3.6 Реализация модуля ConverterModule

3.6.1 Реализация архитектуры ConverterModule

Для реализации описанного фунционала был спроектирован модуль согласно схеме на рисунке 3.8.

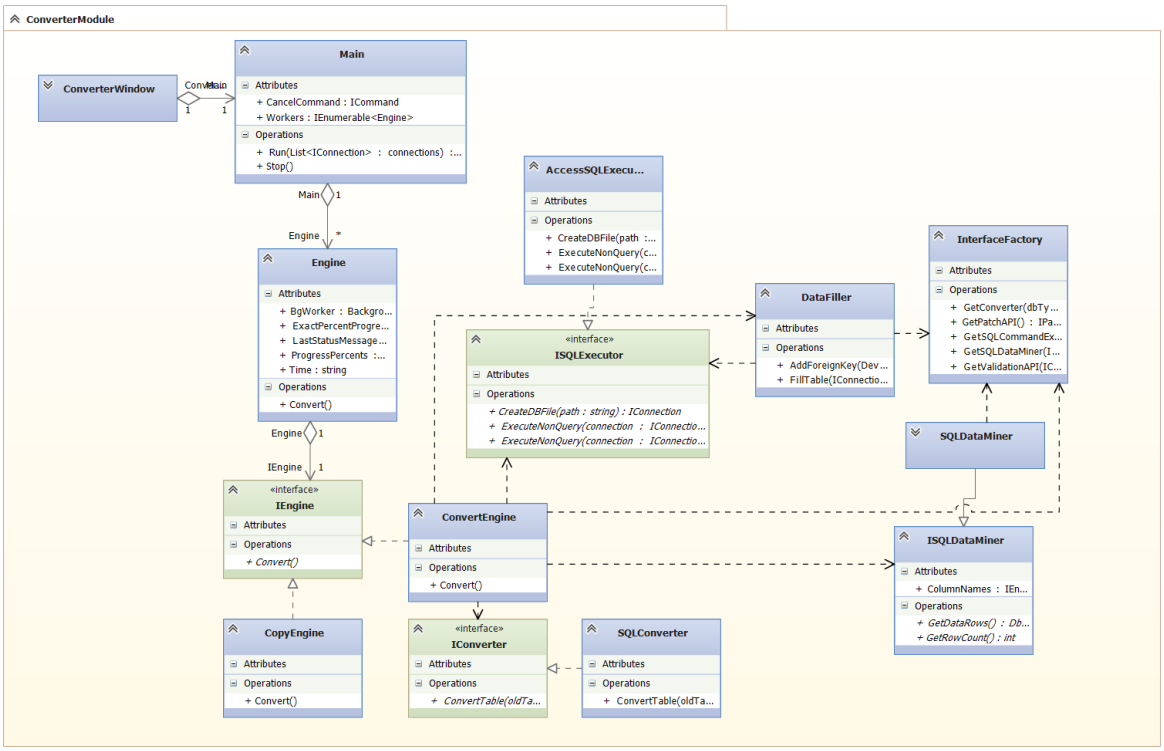


Рисунок 3.8 – Диаграмма классов UML для модуля ConverterModule

При запуске приложения создаётся окно WPF, которое содержит как модель класс Main, являющийся обязательным классом для каждого модуля как точка запуска. Для создания обработчиков конвертации создаются объекты

класса Engine для каждого из соединений. Для обеспечения многопоточности приложения используется компонент BackgroundWorker, являющимся стандартным компонентом разработки. Этот компонент содержит события обработки, завершения работы, остановки, что упрощает процесс настройки потока. В зависимости от типа БД, она может быть скопирована с помощью CopyEngine если MS Access или конвертирована с помощью ConverterModule. [34]

Для создания отчётности о выполнении работы у класса Engine имеются атрибуты, которые согласно паттерну наблюдатель при помощи стандартного интерфейса INotifyPropertyChanged сообщаются в окно состояния конвертации при изменениях значений.

Создание базы данных производится при помощи класса AccessExecutor, который использует библиотеку ADOX, созданную для управления базами данных. [35]

Получение описания схемы базы данных выполняет ValidationModule. Для конвертации полученных моделей используется класс DBObjectsResolver библиотеки DevelopmentLib. Для получения кода SQL для создания схемы базы данных используется модуль PatchModule. Выполнение кода SQL в базе данных MS Access поддерживается при помощи класса AccessSQLExecutor, который может выполнять запросы с возвращением результата и без.

Заполнение данных производится при помощи класса DataFiller, который содержит запросы для вставки данных в базу данных. Перенос остальных объектов производится заново в приведенной последовательности вызова методов классов.

Для получения целостной картины функционирования приложения необходимо привести диаграмму классов всей системы, так как ConverterModule использует практически все компоненты для своей работы (приложение Д).

3.6.2 Реализация интерфейса ConverterModule

Для проектирования представления окна статуса конвертации был создан эскиз окна (рисунок 3.9).

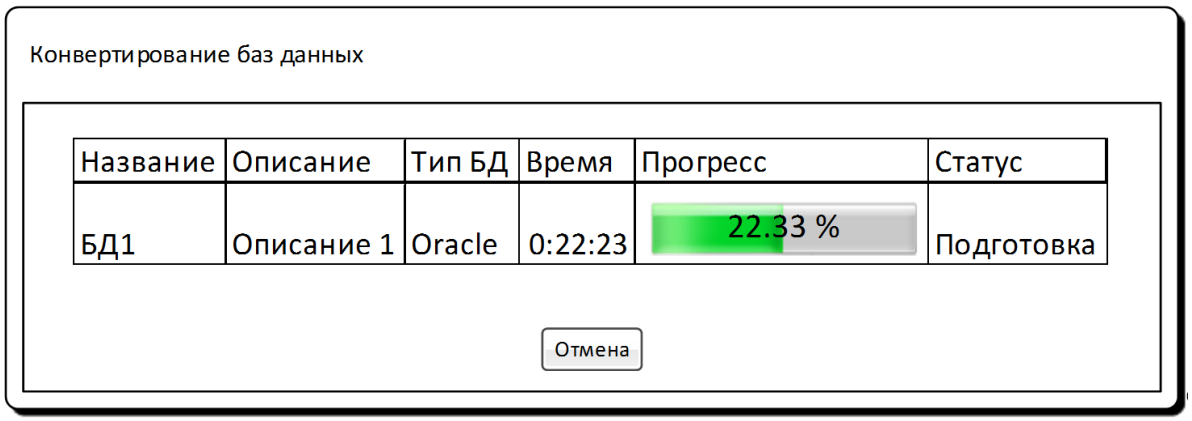


Рисунок 3.9 – Эскиз статуса конвертирования баз данных в виде проволочной диаграммы

Для разработки данного окна также используется WPF и язык разметки XAML в ConverterWindow. Для отображения данных в виде таблицы используется контейнер GridView. Используя в качестве модели класс Main окно получает данные по статусу конвертации каждого из созданного потоков. При нажатии на кнопку «Отмена» конвертация данных приостанавливается. По завершению конвертации кнопка «Отмена» заменяется на кнопку «Ок».

4 Результаты разработки

В результате стадии кодирования все спроектированные интерфейсы и алгоритмы были реализованы, что привело к созданию приложения, осуществляющую конвертацию данных и поддерживающую функции других модулей. К приложению было разработано руководство пользователя (приложение Ж). Для рассмотрения работы приложения был приведен пример работы программы.

При запуске приложения отображается главное окно со списком доступных модулей (рисунок 4.1).

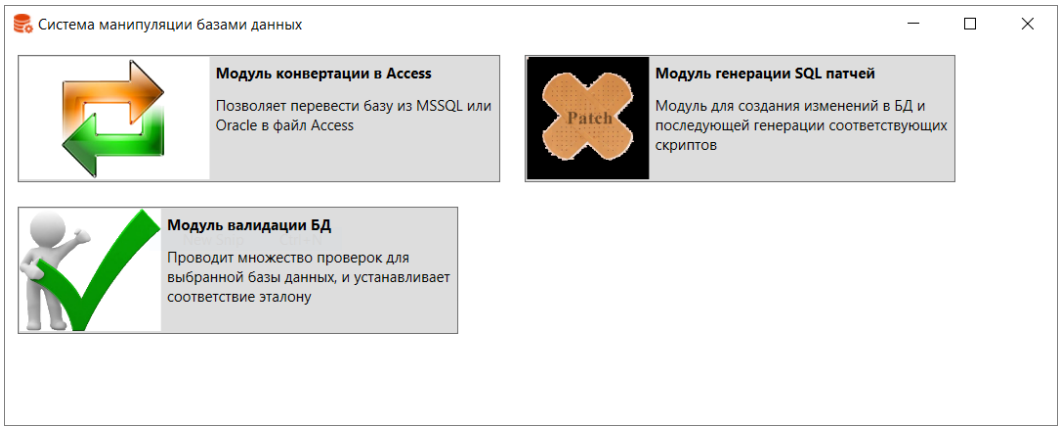


Рисунок 4.1 – Главное окно приложения

Все данные по модулям были успешно загружены и отображены в окне, которое свободно масштабируется, распределяя элементы по окну.

Целью работы являлось создания модуля конвертации с поддержкой других модулей, поэтому только для модуля конвертации существуют дальнейшие окна графического интерфейса, остальные модули поддерживаются на уровне программных интерфейсов.

После нажатия на копку модуля конвертации происходит выбор баз данных для конвертации (рисунок 4.2).

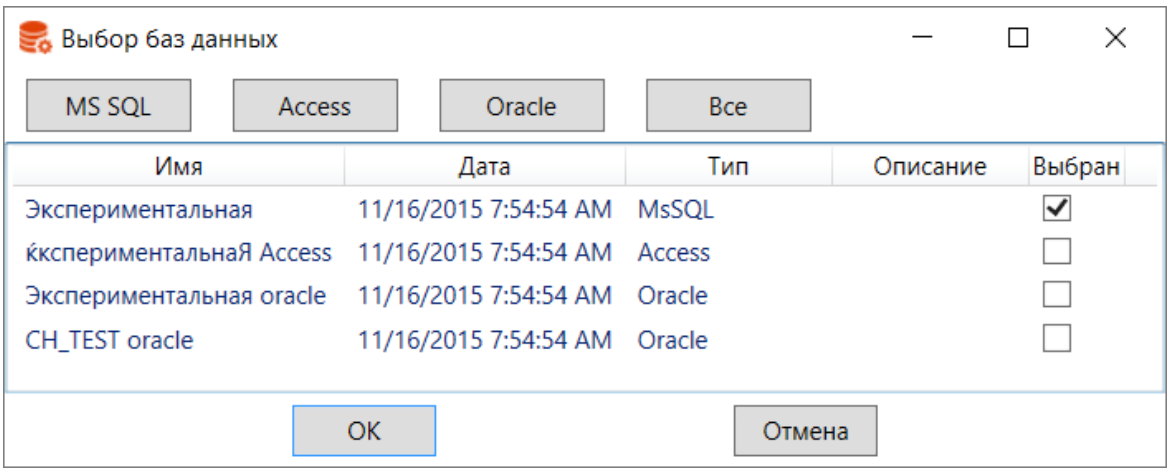


Рисунок 4.2 – Окно выбора баз данных

При нажатии кнопок фильтрации выбираются лишь необходимые типы баз данных. После нажатия на кнопку «ОК» происходит конвертация баз данных в нескольких потоках (рисунок 4.3).

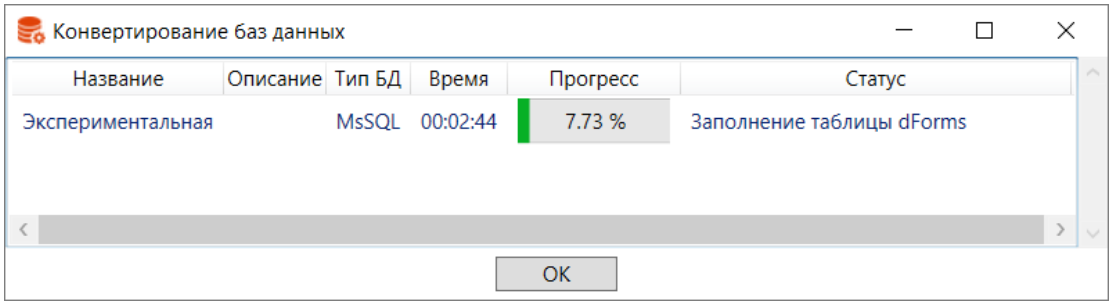


Рисунок 4.3 – Окно статуса конвертации баз данных

Согласно требованиям задания, было разработано многомодульное приложение, осуществляющее поддержку баз данных в рамках предприятия ООО «Химсофт». В дальнейшем для ввода приложения в эксплуатацию необходимо провести тестирование приложения.

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ»

Студенту:

Группа	ФИО
8и2б	Поданеву Сергею Евгеньевичу

Институт	Уровень образования	Кафедра	Направление/специальность
Кибернетики	бакалавриат	Вычислительной техники	09.03.02 Информационные системы и технологии

Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:	
1. *Стоимость ресурсов научного исследования (НИ): материально-технических, финансовых, информационных и человеческих*	*Работа с информацией, представленной в российском законодательстве, экономических материалах.*
2. *Нормы и нормативы расходования ресурсов*	
3. *Используемое федеральное законодательство по отчислениям во внебюджетные фонды, амортизации материальной части.*	
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. *Оценка коммерческого потенциала, перспективности проведения НИ с позиции ресурсоэффективности и ресурсосбережения*	*Оценка потенциальных потребителей исследования, SWOT-анализ.*
2. *Определение ресурсной (ресурсосберегающей), финансовой, бюджетной эффективности исследования*	*Вычисление затрат на проект и экономию в результате реализации.*
Перечень графического материала:	
1. *Матрица SWOT* 2. *График проведения работ*	

Дата выдачи задания для раздела по линейному графику	15.04.2016

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент каф. менеджмента	Николаенко В.С.	–		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8и2б	Поданев Сергей Евгеньевич		

5 Финансовый менеджмент, ресурсоэффективность и ресурсосбережение

5.1 Введение

Коммерческая привлекательность проекта является одним из самых важных объективных критериев важности разработки. Разработка проекта не может быть осуществлена только в целях выполнения разработки. Главной целью является экономический результат. Экономический результат определяется затратами ресурсов на проект, сроком реализации проекта, извлекаемой выгодой.

Для определения экономической целесообразности, перспективности проекта необходимо провести исследования. Целью данного раздела является проектирование и создание конкурентоспособных разработок, отвечающим требованиям в области ресурсоэффективности и ресурсосбережения. Эта цель решается выполнением задач:

- оценкой коммерческого потенциала и перспективности научных иисследований;
- планированием научно-исследовательских работ;
- определением ресурсной, финансовой, экономической эффективности исследования.

5.2 Оценка коммерческого потенциала и переспективости проведения научных исследований с позиции ресурсоэффективности и ресурсосбережения

5.2.1 Потенциальные потребители результатов исследования

Приложение нацелено на обслуживание программного обеспечения организации ООО «Химсофт», поэтому является специализированным. Таким образом потребителями приложения являются сотрудники организации ООО «Химсофт».

5.2.2 SWOT-анализ

SWOT-анализ представляет метод стратегического планирования по анализу внутренней и внешней среды проекта (сильные и слабые стороны, возможности, угрозы). Посредством его выполнения решаются задачи по выявлению возможностей развития проекта, его уязвимостей, что позволяет определить перспективность разработки. SWOT-анализ производится в несколько последовательных этапов. [36]

На первом этапе необходимо сформировать перечисленные факторы внешней и внутренней среды. Результат этого анализа представлен на рисунке 5.1 с включением результата третьего этапа.

Таблица 5.1 – Матрица SWOT

	Сильные стороны: С1. Использование приложения для 3 типов СУБД. С2. Удобство эксплуатации. С3. Отчётность системы на каждом этапе работы.	**Слабые стороны:** Сл.1. Узкая специализация приложения. Сл. 2. Необходимость внесения параметров подключения.

	С4. Высокий уровень поддержки. С5. Простота интерфейса. С6. Настраиваемые источники данных. С7. Возможность добавления функционала. С8. Многопоточное выполнение заданий.	Сл. 3. Избыточная система передачи данных. Сл. 4. Использование приложения только на ОС Windows. Сл. 5. Переход на новую версию языка программирования C# 6.0.
Возможности: В.1. Появление спроса на дополнительный функционал по поддержке БД. В.2. Потребность в выполнении нескольких параллельных задач. В.3. Добавление новых типов данных. В.4. Тенденция производителей СУБД к поддержке языка SQL. В.5. Поддержка новых СУБД использованных поставщиков данных.	Направления развития: В1С4С7 – Добавление новой функциональности. В2С4С8 – Расширение поддержки многопоточности. В3С4С7 – Осуществление поддержки новых типов данных. В4С7 – Расширение поддержки новых типов СУБД В5С4С6 – Введение упрощенной системы добавление СУБД	Сдерживающие факторы: В1Сл1Сл5 – Сложность добавления узкого функционала в новых технологиях. В2Сл5 – Усложнение интерфейса при выполнении нескольких задач одновременно. В5Сл3 – Увеличение избыточности данных для обеспечения универсальности.

Угрозы: У.1. Прекращение поддержки поставщиков данных производителями СУБД. У.2. Переход организации на СУБД одного типа. У.3. Введение инструментов экспорта из каждой СУБД. У.4. Резкое повышение производительности персональных компьютеров. У.5. Сложность использования приложения специалистами без навыков программирования.	Угрозы развития: У1С1С4С6 и У2С1С6 – Устранение поддержки множества баз данных или глубокая переделка на новые поставщики данных. У3С1С2С7 – Перенос части функционала на средства СУБД. У4С2С4 – Часть функционала удобнее реализовать на более требовательных к ресурсам СУБД. У5С2С3С4С5С8 – Угроза сложности использования приложения при усложнении действий и настроек.	Уязвимости: У1Сл2 – Сильная зависимость от поставщиков данных. У2Сл2Сл3 – В случае перехода к СУБД 1 типа ненужная жертва производительностью и простотой интерфейса. У3Сл1Сл3 – В случае введения инструментов экспорта из каждой СУБД не необходима реализация подобных функций. У4Сл3 – Ослабление приложения по сравнению с конкурентами при увеличении производительности ПК, но сохранении производительности приложения. У5Сл1Сл2Сл4 – При введении новых инструментов пользователи могут

		испытывать затруднения в интерфейсе.

Второй этап подразумевает создание интерактивных матриц проекта, в котором возможен анализ сопоставленных параметров матрицы SWOT. Благодаря данному сопоставлению комбинаций возможно получение новых знаний о направлениях развития, сдерживающих факторах, уязвимостях, угрозах развития. Данные интерактивные матрицы представлены в таблицах 5.2 – 5.5.

Таблица 5.2 – Интерактивная матрица проекта сильных сторон и возможностей

Сильные стороны проекта									
Возможности проекта		С1	С2	С3	С4	С5	С6	С7	С8
	В1	-	0	-	+	-	-	+	-
	В2	-	-	-	+	-	-	-	+
	В3	0	-	-	+	-	-	+	-
	В4	0	-	-	-	-	-	+	-
	В5	0	-	-	+	-	+	-	-

Таблица 5.3 – Интерактивная матрица проекта сильных сторон и угроз

Сильные стороны проекта									
Возможности проекта		С1	С2	С3	С4	С5	С6	С7	С8
	У1	+	-	-	+	-	+	-	-
	У2	+	-	-	-	-	+	-	-
	У3	+	+	-	-	-	-	+	-
	У4	-	+	-	+	-	-	-	-
	У5	-	+	+	+	+	-	-	+

Таблица 5.4 – Интерактивная матрица проекта слабых сторон и возможностей

Слабые стороны проекта						
Возможности проекта		Сл1	Сл2	Сл3	Сл4	Сл5
	В1	+	-	-	-	+
	В2	-	-	-	-	+
	В3	-	-	-	-	-
	В4	-	-	-	-	-
	В5	-	-	+	-	-

Таблица 5.5 – Интерактивная матрица проекта слабых сторон и угроз

Слабые стороны проекта						
Возможности проекта		Сл1	Сл2	Сл3	Сл4	Сл5
	У1	-	+	-	-	-
	У2	-	+	+	-	-
	У3	+	-	+	-	-
	У4	-	-	+	-	-
	У5	+	+	-	+	-

На последнем этапе выводятся выводы по результатам интерактивных матриц на основе соответствия или несоответствия параметров анализа. Данные выводы были помещены в таблицу 5.1.

5.3 Планирование научно-исследовательских работ

Планирование работ является одним из первоначальных этапов выполнения работы и необходимо для первоначальной оценки количества работы и распределения ролей и участников проекта по каждому из этапов. Согласно этому плану возможна оценка трудности проекта и разработка графика выполнения работ, которому необходимо следовать в течении выполнения проекта.

Для разработки программного приложения необходимо сначала определиться со структурой работ и её исполнителями, которую затем возможно распланировать на срок создания проекта. Данная структура представлена на таблице 5.6.

Таблица 5.6 – Структура работ

Основные этапы	№ раб	Содержание работ	Должность исполнителя
Разработка технического задания	1	Составление и утверждение технического задания	Руководитель проекта, студент-программист
Выбор направления исследования	2	Подбор и изучение материалов по теме	Студент-программист
	3	Выбор методологии разработки	Руководитель проекта,

			студент-программист
	4	Составление плана работ	Руководитель проекта
Теоритические исследования	5	Выбор средсв разработки	Руководитель проекта, студент-программист
	6	Выбор соглашений по оформлению кода	Руководитель проекта, студент-программист
	7	Выбор паттернов программирования	Руководитель проекта, студент-программист
	8	Подбор средств подключения	Студент-программист
	9	Оценка возможности технологий	Студент-программист
	10	Оценка целесообразности проведения ОКР	Студент-программист
Проведение ОКР			
Проектирование приложения	11	Ознакомление с технологиями предприятия	Студент-программист
	12	Разработка архитектуры	Руководитель проекта, студент-программист
	13	Оценка эффективности архитектуры	Студент-программист
Разработка приложения и разработка технической документации	14	Разработка интерфейса	Студент-программист
	15	Создание кода приложения	Студент-программист
	16	Функциональное тестирование	Студент-программист
	17	Создание руководства пользователя	Студент-программист
Оформление отчёта по НИР	18	Создание пояснительной записки	Студент-программист

Большинство работ запланировано для выполнения студентом-программистом, лишь некоторые контрольные точки выполнения работ выполняются совместно с руководителем. Однако необходимо добавить, что каждый из этапов выполняется под контролем руководителя проекта, чтобы провести работу максимально качественно и приобрести “правильные” навыки по проведению разработки приложения.

5.4 Определение ресурсной, финансовой, экономической составляющей работ

5.4.1 Определения трудоёмкости выполнения работ

Для определения соотношения полезности разработки необходимо также провести анализ трудоемкости работ, так как труд разработчика-программиста является важным ресурсом, возможности которого ограничены.

Определение ожидаемой трудоёмкости работы осуществляется по формуле

$$t_{ожi} = \frac{3t_{\min\ i} + 2t_{\max\ i}}{5} \qquad (1)$$

Где t_{mini} – минимально возможная трудоемкость выполнения заданной i-ой работы, человеко-дни;

t_{maxi}– максимально возможная трудоемкость выполнения заданной i-ой работы, человеко-дни.

Предполагаемая минимальная трудоемкость студента разработчика исходя из 8-часового трудового дня предполагается 20 человеко-дней в оптимистичном варианте и 35 дней в самом пессимистичном, что соответствует минимальной и максимальной трудоемкости. Тогда было подсчитано, что ожидаемая трудоемкость работы составляет 26 человеко-дней.

Для руководителя проекта эти показатели значительно ниже, так как он осуществляет в основном проверку работы. Минимальное и максимальное

значение трудоемкости для руководителя составляет 2 и 6 дней соответственно. Тогда ожидаемая трудоемкость руководителя составляет 3,6 человеко-дня.

Так как разработчиком является один студент, то продолжительность работы в рабочих днях равна выведенной ожидаемой трудоемкости.

Для выполненной схемы работ в таблице 5.6 был составлен календарный план-график выполнения работ (приложение Е). Данный график позволяет сформировать сроки выполнения каждого из этапа работ в течении всего времени выполнения ВКР, распределяя таким образом подсчитанную нагрузку трудоемкости выполнения.

Подсчёт технологической трудоемкости приложения позволяет произвести расчёт в человеко-часах. Данное измерение является абстрактным, для полноты оценки в нынешней экономической ситуации необходимо рассчитать материальные затраты по разработке программного продукта. Помимо затрат на работу программистов необходимо рассчитать затраты на другие материальные ресурсы такие как инструменты разработки.

Для разработки программного приложения в данном проекте необходимо использование персональных компьютеров. Для каждого из принимающих участие в проекте специалистов достаточно наличие одно из них. Таким образом работа выполнялась на 2 комплектах персональных компьютеров стоимостью 47000 руб. и 55000 руб. для студента и руководителя соответственно. В комплект входит системный блок, мышь, клавиатура, монитор. А также использовался принтер стоимостью 27000 руб. Так как помимо приобретения товара необходимо нести расходы по доставке и монтажу, то расходы увеличиваются на 15 %. Итого цена на закупку оборудования равна 148350 руб.

Таблица 5.7 – Материальные затраты на оборудование

Наименование	Единица измерения	Количество		Цена за ед., руб	
		Студент-инженер	Руководитель	Студент-инженер	Руководитель
Персональный компьютер	шт	1	1	54050	63250
Принтер	шт	1		31050	
Итого				85100	63250

Расчётом расходными материалами такими как канцелярские принадлежности и картридж для принтера возможно пренебречь, так как основная часть разработки проходила при помощи использования электронных документов. Только во время разработки пояснительной записки была необходима печать документов. Во время этого этапа использовался лишь небольшая часть ресурса картриджа и количества канцелярских товаров.

Выбранное оборудование использовалось лишь относительно короткий срок по сравнению с нормами использования данного оборудования. После разработки проекта возможно использование данных материальных ресурсов. Для более точной оценки использованных ресурсов необходимо посчитать амортизационные расходы за 5 месяцев разработки.

Амортизационные отчисления за месяц определяются по формуле:

$$З_{ам} = \frac{(Ц_i \cdot Н_a)}{100\ \%},$$

где $Ц_i$ – цена оборудования,

$Н_a$ – норма амортизационных отчислений в %.

Норма амортизационных отчислений зависит от срока полезного использования объектов и равна отношению 100% и срока полезного использования. Этот срок определяется в постановлении правительства РФ. [37]

Принтеры и компьютеры относятся ко второй группе. Следовательно, срок их полезного использования 750 дней. Тогда $Н_a$ = 0.133%. С учетом данных параметров была составлена таблица 5.8, которая вычисляет амортизацию и

конечную стоимость оборудования после завершения проекта в течении 5 месяцев.

Таблица 5.8 – Стоимость оборудования с учетом амортизации

Наименование	Единица измерения	Количество		Изначальная стоимость		Амортизация		Конечная стоимость	
		Студент-инженер	Руководитель	Студент-инженер	Руководитель	Студент-инженер	Руководитель	Студент-инженер	Руководитель
Персональный компьютер	шт	1	1	54050	63250	10810	12650	43240	50600
Принтер	шт	1		31050		6210		24840	
Итого				85100	63250	17020	12650	68080	50600

Таким образом оборудование потеряло стоимость на сумму 29670 руб.

Помимо использования материальных ресурсов были использованы трудовые ресурсы студента-разработчика, руководителя проекта. Данные ресурсы являются ограниченными в рамках определенного промежутка времени и могут быть экономически оценены в виде заработной платы. Пускай студент-разработчик имеет заработную плату по тарифной ставке 20000 руб. в месяц, которая соответствует заработной плате младшего программиста в текущих вакансиях. Данное допущение было сделано, чтобы оценить расходы на разработку проекта и сравнить их с расчётом экономии средств в результате разработки программного приложения. Месячный заработная плата по тарифной ставке руководителя проекта от предприятия равна 50000 руб. Согласно данным цифрам заработной платы и рассчитанной трудоемкости проекта вычислим заработную плату участников проекта.

Заработная плата вычисляется на основе заработной платы по тарифной ставке, премиальных коэффициентов, доплат и надбавок, районных коэффициентов. Общий коэффициент всех надбавок высчитывается по формуле:

$$K = (1 + k_{пр} + k_{д}) * k_{р},$$

где $k_{пр}$ – премиальный коэффициент, равный 0,3 (т.е. 30% от $З_{тс}$);

$k_{д}$ – коэффициент доплат и надбавок, принятый в данной работе за 0, так как отсутствуют вредные производства.

$k_{р}$ – районный коэффициент, равный 1,3 (для Томска)

Для подсчитывания заработной плате пропорционально потраченному времени необходимо рассчитать среднедневную заработную плату. Эта заработная плата рассчитывается по формуле

$$З_{дн} = \frac{З_{м} \cdot М}{F_{д}}$$

где Зм – месячный должностной оклад работника, руб.;

М – количество месяцев работы без отпуска в течение года (для руководителя от предприятия – 11,2 месяца, для студента – 10 месяцев);

Fд – действительный годовой фонд рабочего времени научно-технического персонала, раб. дн. (для научного руководителя – 199 раб. дн., для студента – 186 раб. дн.)

После высчитывания общего коэффициента необходимо также добавить расчёт дополнительной заработной платы, предусмотренных в Трудовом кодексе РФ за отклонение от нормальных условий труда и выплат. Дополнительная заработная плата равна 0.15 от основной заработной платы для проектирования. Результат всех подсчётов представлен в таблице 5.9.

Таблица 5.9 – Таблица посчёта заработной платы исполнителям проекта

Исполнители	Зтс, руб.	kпр	kд	kр	Зм, руб.	Здн,руб	Тр, раб. дн.	Зосн, руб.	Доп. Зар. Плата, руб.	Общая зарплата, руб.
Студент-программист	15000	0.3	0	1.3	25350	1362.90323	26	35435.48	5315.322581	40750.81
Руководитель проекта	50000	0.3	0	1.3	84500	4755.77889	3.6	17120.8	2568.120603	19688.92

Согласно законодательству Российской Федерации, помимо выплаты заработной плате работникам предприятия необходимо производить обязательные отчисления в органы государственного страхования, пенсионный фонд, медицинского страхования. Согласно федеральному закону №212 о страховых взносах назначен тариф страхового взноса в пенсионный фонд РФ 22%, фонд социального страхования РФ 2.9%, федеральный фонд обязательного медицинского страхования 5.1%. Итого страховые взносы составляют 30%, но

для научных и образовательных учреждений они составляют 27,1 %. Итоговая таблица расчётов представлена таблицей 5.10.

Таблица 5.10 – Расчёт отчислений на страховые взносы

Исполнители	Общая зарплата, руб.	Страховые взносы, %	Страховые взносы, руб	Общие затраты, руб.
Студент-программист	40750.81	27.1	11043.469	51794.28
Руководитель проекта	19688.92		5335.6986	25024.62
Итого			16379.167	76818.9

Программа для обслуживания баз данных предприятия необходима, чтобы уменьшить расходы на действия по обслуживанию баз данных. Экономия происходит за счёт уменьшения времени работы специалиста и экономии на персональных компьютерах в результате переноса баз данных на менее требовательную к ресурсам СУБД MS Access.

В предприятии на текущий момент закупаются ноутбуки ценой 51000 руб. для обслуживания всех типов баз данных. Для обслуживания баз данных СУБД MS Access необходимы ноутбуки ценой 32000 руб. Также в результате различных факторов в процессе эксплуатации данные ноутбуки меняются на предприятии раз в год. С учетом данных параметров возможен подсчёт экономии материальной части в год (таблица 5.11).

Таблица 5.11 – Экономия материальной части

Оборудование	Стоимость, руб.	Расходы на траспорт, %	Периодичность обновления, лет	Общая стоимость за год, руб.
Ноутбук до введения приложения	51000	15	1	58650
Ноутбук после введения приложения	31000			35650
Экономия				23000

Программное приложение также позволяет экономить трудовые ресурсы для выполнения операций по поддержке баз данных. На предприятии в среднем совершается 5 обращений к базам данных при помощи средств СУБД. В среднем на данные операции уходит 4 часа по каждому обращению. Посредством автоматизации этих операций планируется сократить время обслуживания человеком до 10 мин. Используя посчитанные отчисления на руководителя проекта от предприятия высчитаем экономию средств (таблица 5.12).

Таблица 5.12 – Экономия трудовых ресурсов

	Заработная плата со страховыми взносами за месяц, руб.	Среднедневная плата, руб.	Периодичность обращений, раз в месяц	Время обращения, час.	Стоимость за год, руб.
До введения приложения	109850	6182.512563	5	4	185475.377
После введения приложения				0.17	7728.1407
Экономия					177747.236

5.4.2 Подсчёт экономической эффективности проекта

Подсчитанные затраты и экономия средств в результате выполнения проекта позволяют получить экономическое обоснование целесообразности выполнения проекта.

Совокупная стоимость проекта с учетом затрат на оборудование и трудовые ресурсы составляет 106488.9 руб. Данная сумма вычислялась за счёт амортизационной потери стоимости оборудования во время выполнения проекта, суммарных выплат работникам и обязательным фондам.

Экономия проекта составила 200747.24 за счёт экономии средств на закупку оборудования и экономии рабочего времени работников.

Итого проект позволяет только за год сэкономить 94258.34 руб., что является главным показателем необходимости создания программного обеспечения.

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Студенту:

Группа	ФИО
8и2б	Поданеву Сергею Евгеньевичу

Институт		Кафедра	
Кибернетики	бакалавриат	Вычислительной техники	09.03.02 Информационные системы и технологии

Исходные данные к разделу «Социальная ответственность»:	
1. Характеристика объекта исследования и области его применения	*Объектом исследования являются алгоритмы обработки баз данных, рабочее место при разработке. Программное обеспечение планируется использовать в области поддержки баз данных.*
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Производственная безопасность 1.1. Анализ выявленных вредных факторов при разработке и эксплуатации проектируемого решения. 1.2. Анализ выявленных опасных факторов при разработке и эксплуатации проектируемого решения.	Анализ вредных факторов: 1. Повышенная или пониженная температура воздуха рабочей зоны 2. Повышенный или пониженный уровень отрицательных и положительных аэроионов 3. Повышенная напряженность электрического поля 4. Отсутствие или недостаток естественного света 5. Недостаточная искусственная освещенность рабочей зоны 6. Повышенная яркость света 7. Повышенная контрастность 8. Прямая и отраженная блесткость 9. Повышенный уровень шума на рабочем месте Анализ опасных факторов: 1. Повышенный уровень статического электричества 2. Повышенное значение напряжения в электрической цепи, замыкание которой может произойти через тело человека
2. Экологическая безопасность	Проведен анализ отходов, образующихся в процессе работы,
3. Безопасность в чрезвычайных ситуациях	Рассмотрена наиболее вероятная чрезвычайная ситуация - пожар
4. Правовые и организационные вопросы обеспечения безопасности	Приведен анализ специальных правовых норм, рабочего места сидя и в особенности при работе с ПЭВМ.

Дата выдачи задания для раздела по линейному графику	15.04.2016

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент каф. ЭБЖ	Мезенцева И.Л.	-		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8и2б	Поданев Сергей Евгеньевич		

6 Социальная ответственность

Проект ВКР заключается в разработки программного обеспечения по верификации и адаптации баз данных предприятия ООО «Химсофт». Выполнение данной работы позволит экономить трудовые и материальные ресурсы по обслуживанию баз данных различных типов. Таким образом ввиду высокой специализации программного обеспечения для предприятия, оно планируется к использованию только в этой организации и организациям, приобретающим продукты данной организации. Автоматизация процессов администрирования баз данных позволит использовать данную программу широкому кругу пользователей: программистам, химикам-аналитикам, которые выполняют свою работу на персональных компьютерах.

В данном разделе необходимо рассмотреть вопросы производственной, экологической безопасности, безопасности в чрезвычайных ситуациях, правовые и организационные вопросы обеспечения безопасности. Данные пункты необходимо рассмотреть, так как деятельность неизбежно влияет на окружающую среду и выполняется в при некоторых условиях, которые необходимо учитывать. Неисполнение этих условий влечёт угрозу безопасности окружающей среде, работникам и даже самой деятельности.

Для анализа вопросов социальной ответственности необходимо рассмотреть рабочее место оператора персонального компьютера, опасности, которые может принести эксплуатация ЭВМ, и меры защиты от них.

6.1 Профессиональная социальная безопасность

В данной части работы необходимо выявить источники опасности при разработке программного обеспечения и его эксплуатации.

Согласно ГОСТ 12.0.003-74[38] опасные факторы подразделяются на физические, химические, биологические и психофизические. Из перечисленных групп необходимо выбрать факторы, свойственные при работе с ЭВМ. Опасности работы с персональным компьютером как правило связаны со

статическими физическими нагрузками и психофизическими факторами, что связано с безопасностью работы с электрооборудованием и в частности с монитором компьютера, условиями организации производственной среды. При выполнении работ на персональном компьютере(ПК) были выделены опасные и вредные факторы в таблице 6.1.

Таблица 6.1 – Вредные и опасные факторы при работе с ПК

Источник фактора, наименование видов работ	Факторы (по ГОСТ 12.0.003-74)		Нормативные документы
	Вредные	Опасные	
1.Работа за ПК	1. Повышенная или пониженная температура воздуха рабочей зоны	10. Повышенный уровень статического электричества	СанПиН 2.2.4-96[39]/ГОСТ 12.1.030-81[40]
	2. Повышенный или пониженный уровень отрицательных и положительных аэроионов	11. Повышенное значение напряжения в электрической цепи, замыкание которой может произойти через тело человека	СанПиН2.2.4.1294-03 [41]/ГОСТ 12.1.038-82[42]
	3. Повышенная напряженность электрического поля		СанПиН 2.2.2/2.4.1340-03[43]
	4. Отсутствие или недостаток естественного света		СаНПиН 2.2.2/2.4.1340-03
	5. Недостаточная искусственная освещенность рабочей зоны		
	6. Пониженная яркость света		
	7. Повышенная контрастность		

	8. Прямая и отраженная блесткость		
	9.Повышенный уровень шума на рабочем месте		СанПиН 2.2.2/2.4.1340-03, СН2.2.4/2.1.8.562-96[44]

Для защиты от данных вредных и опасных факторов необходимо их проанализировать - выявить источники возникновения, нормы воздействия.

6.1.1 Анализ выявленных вредных факторов при разработке и эксплуатации проектируемого решения

6.1.1.1 Повышенная или пониженная температура рабочей зоны и повышенная температура рабочих поверхностей

В СанПиН 2.2.4-96 «Гигиенические требования к микроклимату производственных помещений», определяются параметры микроклимата рабочих мест:

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

Эти показатели возникают вследствие воздействия нескольких факторов на микроклимат: работа оборудования, устройство производственных помещений, времени года. Например, температура воздуха зависит от количества тепловой энергии, выделяемой техникой, температуры внешней среды, средств вентиляции и отопления. Эти параметры микроклимата влияют на функциональное состояние, самочувствие, работоспособность и здоровье людей.

При выборе показателей микроклимата необходимо выбирать степень тяжести работ, так как от рода деятельности зависят допустимые нагрузки на человека. Проводимая работа относится к работе разработчика-программиста и относится к категории тяжести 1а. Тогда возможно определить допустимые и оптимальные величины показателей микроклимата (таблица 6.2 и 6.3).

Таблица 6.2 - Допустимые величины показателей микроклимата

Период года	Температура, °С	Относительная влажность, %	Скорость движения воздуха, м/с
Холодный	20-25	15-75	0,1-0,2
Тёплый	21-28	15-75	0,1-0,2

Таблица 6.3 - Оптимальные величины показателей микроклимата

Период года	Температура, °С	Относительная влажность, %	Скорость движения воздуха, м/с
Холодный	22-24	60-40	0,1
Тёплый	23-25	60-40	0,1

Микроклимат рабочего места должен регулироваться центральным отоплением и проточно-вытяжной вентиляцией. Для улучшения показателей микроклимата необходимо обеспечить правильное кондиционирование и вентиляцию воздуха, отопление помещений. В целях профилактики могут быть использованы средства местного кондиционирования воздуха, воздушного душирования, спецодежда, помещения для отдыха и обогревания, регламентация времени работы, увлажнения воздуха. [39]

6.1.1.2 Повышенный или пониженный уровень отрицательных и положительных аэроионов

В процессе работы приборов, воздействия солнечной радиации, космического излучения образуются одновременно положительные и отрицательно заряженные ионы, называемые аэроионами. Атмосферный воздух

всегда содержит некоторое количество этих частиц в силу естественных процессов. При работе персональных компьютеров, а в основном дисплея, выделяется дополнительное их количество, производится излучение человека. При воздействии избыточного количества частиц на человека могут возникнуть отрицательные последствия, влияя на физико-химические процессы в организме.

Согласно СанПиН2.2.4.1294-03 воздухе должно присутствовать следующее количество отрицательных и положительных аэроионов: минимально 400 положительных и 600 отрицательных ионов, оптимально 3000-5000 положительных и 1500-3000 отрицательных ионов и максимально допустимых 50000 ионов в 1 $см^3$ воздуха. [41]

Для улучшения воздуха могут быть применены ионизаторы и деионизаторы воздуха, которые могут создать искусственно необходимую концентрацию ионов в воздухе.

6.1.1.3 Повышенная напряженность электрического поля

Электрическое поле создаётся зарядами. Как при действии любой из техники под действием токов создаётся электрическое поле. Наибольшее его значение сосредоточено в блоке питания, так как в нем осуществляется преобразование тока 220В в токи малого напряжения, которые создают меньшие значения электромагнитных полей. Для уменьшения воздействия блоки питания экранируются.

При повышенных уровнях напряженности электрического поля развиваются нарушения нервной, сердечно-сосудистой системы.

Согласно СанПиН 2.2.2/2.4.1340-03 временные допустимые уровни электромагнитного излучения выражены в таблице 6.4. [43]

Таблица 6.4 - Временные допустимые уровни ЭМП, создаваемых ПЭВМ на рабочих местах

Наименование параметров		ВДУ ЭМП
	в диапазоне частот 5 Гц - 2 кГц	25 В/м

Напряженность электрического поля	в диапазоне частот 2 кГц - 400 кГц	2,5 В/м
Плотность магнитного потока	в диапазоне частот 5 Гц - 2 кГц	250 нТл
	в диапазоне частот 2 кГц - 400 кГц	25нТл
Напряженность электростатического поля		15 кВ/м

Для защиты от электромагнитного излучения необходимо использовать оборудование с меньшим излучением, удалить системный блок и монитор как можно дальше, сократить время работы и вводить перерывы, заземлить компьютер.

6.1.1.4 Отсутствие или недостаток естественного света, недостаточная искусственная освещенность рабочей зоны, повышенная яркость света, пониженная контрастность, прямая и отраженная блесткость

Освещённость рабочей зоны влияет на работу глаз. Неправильное задание освещенности пагубно влияет на глаза, обеспечивая излишнее напряжение на нервную систему в результате усилий на обработку менее четких сигналов. В результате периодическое напряжение приводит к ослаблению зрения. При этом установлено, что наилучшей освещённостью является естественный свет. Именно поэтому необходимо установить искусственный свет таким образом, чтобы он максимально имитировал естественный. Это достигается путем настройки местного и общего освещения.

Согласно СаНПиН 2.2.2/2.4.1340-03 освещенность поверхности стола в зоне размещения документа должна быть 300-500 лк. Освещенность поверхности экрана должна быть не более 300 лк и освещение не должно создавать бликов на экране. Прямая блескость от источников освещения должна быть не более 200 кд/м^2. Отраженная блескость не должна превышать 40 кд/м^2.

Соотношение яркости между рабочими поверхностями не должно превышать 3:1 – 5:1, между рабочими поверхностями и поверхностями стен и оборудования 10:1. Рабочие столы необходимо размещать таким образом, чтобы естественный свет падал преимущественно слева, а коэффициент естественной освещенности не должен превышать 0.5%. Коэффициент пульсации не должен превышать 5 %. [41]

Для минимизаций воздействия плохой освещенности возможно введение дополнительного местного освещения, уменьшения работы в неблагоприятных условиях, увеличение времени работы в дневное время суток при естественном освещении.

6.1.1.5 Повышенный уровень шума на рабочем месте

Помимо перечисленных вредных факторов, одним из них является уровень шума на рабочем месте. При повышенном уровне становится трудно разобрать речь и сигналы, повышается утомляемость и ухудшается состояние здоровье. При регулярном нарушении уровня шума болезни принимают хронический характер, а чувствительность слуховой системы понижается. Шум действует не только на слуховую систему, но на нервную систему в целом, что приводит к проблемам памяти, концентрации, реакции. В итоге это приводит к понижению производительности работников.

Согласно СанПиН 2.2.2/2.4.1340-03 шумящее оборудование, уровни шума которого превышают нормативные, должно размещаться вне помещений с ПЭВМ. Согласно санитарным нормам СН2.2.4/2.1.8.562-96 для творческой деятельности в т.ч. программирования установлен предельный уровень звука 50 дБА. [41] [44]

6.1.2 Анализ выявленных опасных факторов при разработке и эксплуатации проектируемого решения

6.1.2.1 Статическое электричество

Статическое электричество является одним из опасных факторов. Оно возникает в результате накопления заряда на поверхности или объеме диэлектриков или на изолированных проводниках. Опасность возникает при разряде этого заряда, поскольку заряд может быть велик, что может вызвать поломку оборудования, угрозу для жизни оператора в результате прохождения заряда через тело человека. Накопление заряда может быть вызвано изоляцией оборудования или оператора.

Для устранения необходимо обеспечить заземление компьютера, чтобы заряд мог уходить на землю, не накапливаясь на корпусе компьютера. Правила заземления и другие мероприятия по устранению статического электричества перечислены в ГОСТ 12.1.030-81. [40]

6.1.3 Повышенное значение напряжения в электрической цепи

При прикосновении к электроустановкам, по которым протекает ток возможно прохождение тока через тело человека, так как земля имеет нулевой потенциал. Сила поражения завит от времени воздействия, характера тока, состояния человека, места соприкосновения, пути прохождения тока. Поражающими факторами являются ожоги, сбои в работе сердечной мышцы, нервной системы. Причиной прикосновения является как правило нарушение изоляции изделия и нарушение техники безопасности.

Для питания персональных компьютеров в России как правило используется сетевое напряжение 230В переменного тока при частоте 50 Гц. После прохождения через блок питания ток трансформируется в постоянный ток напряжения 5 и 12 В. Согласно ГОСТ 12.1.038-82 допустимо прохождение через тело человека переменного тока с напряжением 2В, силой 0.3мА, постоянного тока 8В, силой 1мА. Очевидно, прикосновение к проводке компьютера в любой

из его частей недопустимо, хотя после преобразования в блоке питания может не нести поражающих действий. [42]

Для защиты от прохождения тока по телу необходимо соблюдать технику безопасности использования персонального компьютера, использовать ГОСТ Р 50571.3-94 по защите от поражения электрического тока. При обнаружении нарушений изоляции необходимо в первую очередь отключить прибор от источника питания, использовать изоляционные материалы. [45]

6.2 Экологическая опасность

В процессе работы персональные компьютеры оказывают вредное воздействие на окружающую среду. Воздействие заключается в создании вредного излучения, сжигании кислорода, создании вибраций, загрязнении вредными отходами деятельности. Для обеспечения экологической безопасности необходимо рассмотреть воздействие и обеспечить защиту селитебной зоны, атмосферы, гидросферы и литосферы.

6.2.1 Защита литосферы

При выполнении работы на персональных компьютерах, некоторые из частей компьютера могут прийти в негодность или устареть, тогда необходимо их утилизировать. В качестве отходов в результате деятельности программирования может выступать бумага, отходы электроники, ртутьсодержащие лампы. Эти твердые бытовые отходы загрязняют окружающую среду и их необходимо перерабатывать, чтобы максимально снизить их вредное воздействие. Для складирования мусора перед утилизацией предлагается вывоз на территории, предназначенные для этого. Утилизация электроники производится специальными компаниями и является обязательной, в особенности утилизация ртутьсодержащих ламп. Отработанные ртутьсодержащие лампы подлежат проведению демеркуризации.

6.3 Безопасность в чрезвычайных ситуациях

В процессе работы на персональном компьютере единственной чрезвычайной ситуацией, которую создаёт компьютер может стать пожар. Пожар в течении работы персонального компьютера может возникнуть в результате замыкания электропроводки, приводящее к возгоранию аппаратуры. Далее огонь может распространиться на другие объекты рабочего помещения, создавая техногенную чрезвычайную ситуацию.

Федеральный закон №123 "Технический регламент о требованиях пожарной безопасности" устанавливает основные принципы установления пожарной безопасности. Для предупреждения возникновения пожаров необходимо прежде всего следовать технике безопасности пользования персональным компьютером. Приведенный технический регламент предусматривает предотвращение пожаров путем исключения условий образования горючей среды (статья 49) и исключений условий образования в горючей среде источников зажигания (статья 50). [46]

Исключение условий образования горючей среды в сложившихся рабочих условиях включает в себя: применение минимального количества горючих веществ и материалов и наиболее безопасного их размещения, изоляция источников зажигания от горючих веществ, понижение концентрации окислителя в горючей среде в защищаемом объеме, удаление из помещений пожароопасных отходов производства, отложений пыли и пуха. Исключение источников зажигания достигается путем применения оборудования, исключающих образование статического электричества, устройство молниезащиты зданий, сооружений и оборудования, применение устройств, исключающих возможность распространения пламени из одного объема в смежный.

При возникновении пожара необходимо обеспечить технические и организационные мероприятия по оповещению, эвакуации людей, тушению

пожара. При обнаружении пожара необходимо оповестить пожарную службу посредством пожарной сигнализации и провести эвакуацию через эвакуационные пути и выходы. Ликвидация последствий пожара обеспечивается автоматическими установками пожаротушения и пожарными службами.

6.4 Правовые и организационные вопросы обеспечения безопасности

Правовые и организационные вопросы задают специфичные требования для выбранного рода деятельности, а именно программирования. Для учёта всех требований законодательства по организации работы необходимо провести анализ специальных норм трудового законодательства и организационных мероприятий по компоновке рабочей зоны для обеспечения эффективности работы трудящихся.

6.4.1 Специальные правовые нормы трудового законодательства

Специальные ограничения работы специалиста – программиста практически ограничиваются общими нормами трудового законодательства и СаНПиН 2.2.2/2.4.1340-03.

Согласно 12 пункту СаНПиН 2.2.2/2.4.1340-03 женщины со времени установления беременности допускаются к работе на ПЭВМ до 3 часов за рабочую смену. Лица работающие с ПЭВМ более 50% рабочего времени должны проходить обязательные предварительные медицинские осмотры. [43]

Согласно постановлению главного государственного санитарного врача РФ от 03.06.2003 № 118 продолжительность непрерывной работы без регламентированного перерыва должна быть меньше 1 часа. В зависимости от категории работы с ПЭВМ уровень нагрузки за рабочую смену может доходить до 6 часов с суммарным временем регламентированным временем перерывов от 50 мин. до 140 мин. В зависимости от продолжительности смены. Рекомендуется организовывать перерывы на 10-15 мин. через каждые 45-50 мин. работы. [47]

6.4.3 Организационные мероприятия по компоновке рабочей зоны

Организационные мероприятия по компоновке рабочей зоны заключаются в правильном расположении персонального компьютера относительно оператора и правильном положении самого оператора во время работы. Рабочее зона, в том числе и рабочее место должна обеспечивать наивысшую эффективность работы оператора с минимальными нагрузками на здоровье человека. При несоблюдении условий работник может получить производственные травмы и хронические заболевания.

Большинство мероприятий регулируются 10 пунктом СаНПиН 2.2.2/2.4.1340-03 “Требования к организации и оборудованию рабочих мест с ПЭВМ для взрослых пользователей”. В этом документе предусматриваются правила расположения высоты рабочей поверхности, рабочего стола, параметры конструкции рабочего стула по ширине, глубине, высоте, краях, углах наклона, подлокотниках. Данные параметры должны регулироваться для обеспечения комфорта работника. Помимо этого, рабочее место необходимо оборудовать подставкой для ног определенных параметров, клавиатуру как главный инструмент работника располагать определенным образом. [43]

Более подробно сидячее рабочее место определяется в ГОСТ 12.2.032-78. Этот документ описывает размерные характеристики рабочего места, требования к размещению органов управления, требования к размещению средств отображения информации. [48]

При следовании этим документам необходимо также учитывать размеры программиста, которые могут влиять на некоторые из заданных величин, которые были выведены по среднестатистическим показателям.

Следование установленным правилам и дополнительным медицинским рекомендациям, выполнением специальных физических упражнений, снижающих нагрузку, позволит избежать травм и болезней, увеличить производительность работников, предупреждая раннее утомление.

Заключение

В ходе выполнения выпускной квалификационной работы была достигнута цель выполнения работы – создание многомодульного приложения по адаптации и верификации баз данных. Были решены задачи:

- проведен анализ предметной области;
- проведен анализ средств выполнения проекта;
- выполнено проектирование приложения;
- было реализовано приложение;
- проведен анализ вопросов социальной ответственности;
- проведена оценка проекта с точки зрения финансового менеджмента.

Полученное приложение выполняет функциональные требования – поддерживает конвертацию баз данных в базу данных с СУБД MS Access и содержит возможность добавления дополнительных модулей по обеспечению поддержки баз данных предприятия. Цель была достигнута путём применения последовательных этапов с глубоким анализом по разработке приложения.

Решение вопросов социальной ответственности и оценка с точки зрения финансового менеджмента позволила описать целесообразность и возможность создания проекта, а также предупредить опасное воздействие в результате деятельности.

Путем выполнения всех аспектов деятельности был создан программный продукт, который планируется использовать в составе организации ООО «Химсофт».

Список использованных источников

1. A. Meier. Hierarchical to relational database migration // IEEE Software.– 1994.– ISSN :0740-7459.– С.21-27

2. Enterprise Database Architecture Migration [Электронный ресурс]. URL: https://www.inf.ed.ac.uk/publications/thesis/online/IM040132.pdf, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

3. A. Sutcliffe, I. Bennett, A. Doubleday, M. Ryan. Designing Query Support for Multiple Databases // Springer US.– 1995.– ISSN : 1868-4238.– С.207-212

4. A. Achpal, V. B. Kumar, K. Mahesh. Modeling Ontology Semantic Constraints in Relational Database Management System // IMECS.– 2016.– ISSN : 2078-0958.– С.163-168

5. Рейтинг реляционных баз данных [Электронный ресурс]. URL: http://db-engines.com/en/ranking/relational+dbms, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

6. ООО «Химсофт» [Электронный ресурс]. URL: http://www.chemsoft.ru/, свободный. – Яз. рус. Дата обращения: 01.05.2016г.

7. Описание СУБД MS Access [Электронный ресурс]. URL: https://products.office.com/en-us/access, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

8. Описание MS SQL Server [Электронный ресурс]. URL: https://www.microsoft.com/en/server-cloud/products/sql-server/, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

9. Описание СУБД Oracle [Электронный ресурс]. URL: http://www.oracle.com/ru/database/overview/index.html, свободный. – Яз. рус. Дата обращения: 01.05.2016г.

10. Сравнение СУБД MS Access, Oracle и MS SQL Server [Электронный ресурс]. URL: http://db-

engines.com/en/system/Microsoft+Access%3BMicrosoft+SQL+Server%3BOracle, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

11. Методологии разработки программного обеспечения [Электронный ресурс]. URL: http://compress.ru/article.aspx?id=11321, свободный. – Яз. рус. Дата обращения: 01.05.2016г.

12. Методологии разработки программного обеспечения Agile [Электронный ресурс]. URL: http://www.codeproject.com/Articles/604417/Agile-software-development-methodologies-and-how-t, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

13. Стандарт оформления кода [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/Стандарт_оформления_кода, свободный. – Яз. рус. Дата обращения: 01.05.2016г.

14. Стандарт оформления кода команды RSDN [Электронный ресурс]. URL: https://rsdn.ru/article/mag/200401/codestyle.XML, свободный. – Яз. рус. Дата обращения: 01.05.2016г.

15. Дастин Босуэлл, Тревор Фаучер, Читаемый код или программирование как искусство: Пер. с англ. / Дастин Босуэлл, Тревор Фаучер. – СПб: ООО «ДиаСофтЮП», 2012.– 208 с.

16. Сравнение систем управления разработкой программного обеспечения [Электронный ресурс]. URL: https://en.wikipedia.org/wiki/Comparison_of_project_management_software, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

17. Visual Studio Online [Электронный ресурс]. URL: https://www.visualstudio.com/en-us/products/what-is-visual-studio-online-vs.aspx, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

18. Введение в язык C# [Электронный ресурс]. URL: https://msdn.microsoft.com/en-us/library/z1zx9t92.aspx, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

19. Разработка клиентских приложений [Электронный ресурс]. URL: https://msdn.microsoft.com/en-us/library/54xbah2z(v=vs.110).aspx, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

20. ADO.NET для ADO программиста [Электронный ресурс]. URL: https://msdn.microsoft.com/en-us/library/ms973217.aspx, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

21. Поставщики данных ADO.NET [Электронный ресурс]. URL: https://msdn.microsoft.com/en-us/library/dd363565.aspx, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

22. Поставщики данных .Net [Электронный ресурс]. URL: https://msdn.microsoft.com/en-us/library/a6cd7c08.aspx, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

23. Поставщик данных Oracle для .Net [Электронный ресурс]. URL: http://www.oracle.com/technetwork/topics/dotnet/index-085163.html, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

24. Alexander Shvets, Design Patterns Explained Simply, Яз. Англ / Alexander Shvets . – sourcemaking.com, 2015.– 117 с.

25. XmlSerializer [Электронный ресурс]. URL: https://msdn.microsoft.com/en-us/library/system.xml.serialization.xmlserializer(v=vs.110).aspx. – Яз. англ. Дата обращения: 01.05.2016г.

26. Assembly Class [Электронный ресурс]. URL: https://msdn.microsoft.com/en-us/library/system.reflection.assembly(v=vs.110).aspx, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

27. MVVM паттерн [Электронный ресурс]. URL: https://msdn.microsoft.com/en-us/library/hh848246.aspx, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

28. WrapPanel class [Электронный ресурс]. URL: https://msdn.microsoft.com/en-

us/library/system.windows.controls.wrappanel(v=vs.110).aspx, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

29. IValueConverter Interface [Электронный ресурс]. URL: https://msdn.microsoft.com/en-us/library/system.windows.data.ivalueconverter(v=vs.110).aspx, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

30. GridView class [Электронный ресурс]. URL: https://msdn.microsoft.com/en-us/library/system.windows.controls.gridview(v=vs.110).aspx, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

31. ICommand interface [Электронный ресурс]. URL: https://msdn.microsoft.com/en-us/library/system.windows.input.icommand(v=vs.110).aspx. – Яз. англ. Дата обращения: 01.05.2016г.

32. Information Schema Views [Электронный ресурс]. URL: https://msdn.microsoft.com/en-us/library/ms186778.aspx, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

33. Oracle system tables [Электронный ресурс]. URL: https://docs.oracle.com/cd/E21901_01/timesten.1122/e21644/systemtables.htm#TTSYS355, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

34. BackgroundWorker class [Электронный ресурс]. URL: https://msdn.microsoft.com/en-us/library/system.componentmodel.backgroundworker(v=vs.110).aspx, свободный. – Яз. англ. Дата обращения: 01.05.2016г.

35. ADOX API [Электронный ресурс]. URL: https://msdn.microsoft.com/en-us/library/ms681520(v=vs.85).aspx. – Яз. англ. Дата обращения: 01.05.2016г.

36. SWOT анализ [Электронный ресурс]. URL: http://www.stplan.ru/articles/theory/swot.htm, свободный. – Яз. рус. Дата обращения: 01.05.2016г.

37. Постановление Правительства РФ от 1 января 2002 г. N 1 «О Классификации основных средств, включаемых в амортизационные группы» [Электронный ресурс]. URL: http://base.garant.ru/12125271/#block_1000, свободный. – Яз. рус. Дата обращения: 01.05.2016г.

38. ГОСТ 12.0.003-74 «ССБТ. Опасные и вредные производственные факторы. Классификация»

39. СанПиН 2.2.4-96 «Гигиенические требования к микроклимату производственных помещений»

40. ГОСТ 12.1.030-81 «Электробезопасность. Защитное заземление, зануление»

41. СанПиН2.2.4.1294-03 «Гигиенические требования к аэроионному составу воздуха производственных и общественных помещений»

42. ГОСТ 12.1.038-82 «Система стандартов безопасности труда. Электробезопасность. Предельно допустимые значения напряжений прикосновения и токов»

43. СанПиН 2.2.2/2.4.1340-03 «Гигиенические требования к персональным электронно-вычислительным машинам и организации работы»

44. СН2.2.4/2.1.8.562-96 «Шум на рабочих местах, в помещениях жилых, общественных зданий и на территории жилой застройки»

45. ГОСТ Р 50571.3-94 «Требования по обеспечению безопасности. Защита от поражений электрическим током.»

46. Федеральный закон №123 «Технический регламент о требованиях пожарной безопасности» [Электронный ресурс]. URL: http://www.consultant.ru/document/cons_doc_LAW_78699/. – Яз. рус. Дата обращения: 01.05.2016г.

47. Постановление Главного государственного санитарного врача РФ от 03.06.2003 № 118 [Электронный ресурс]. URL: http://www.consultant.ru/document/cons_doc_LAW_42836/, свободный. – Яз. рус. Дата обращения: 01.05.2016г.

48. ГОСТ 12.2.032-78 «Рабочее место при выполнении работ сидя. Общие эргономические требования.»

Приложение А

Техническое задание

1 Введение

1.1 Наименование продукта

Приложение “DBManagement”.

1.2 Краткая характеристика области применения

Программа ориентирована на пользователей компании ООО «Химсофт» и преследует специальные цели по обслуживанию баз данных.

2 Основание для разработки

2.1 Документ, на основании которого ведется разработка

Приказ темы ВКР группы 8и2б.

2.2 Организация, утвердившая документ

Томский политехнический университет.

2.3 Назначение разработки

Приложение позволяет осуществлять поддержку множества удаленных баз данных под управлением 3 типов СУБД: MS SQL, Oracle и MS Access. Путем реализации приложения производиться перенос данных в СУБД MS Access для уменьшения требуемых ресурсов СУБД, автоматизация деятельности по поддержке баз данных для экономии трудовых ресурсов.

3 Требования к программному продукту

3.1 Требования к функциональным характеристикам

1. Приложение должно поддерживать создание SQL модификаций.

- SQL модификация представляет собой SQL код и создаётся на основании входящей модели данных, типа СУБД.

2. Приложение должно поддерживать управление и запуск модулей системы.

- Количество модулей заранее не известно и загружается в виде библиотек файлов формата .dll из локальной папки с приложением;

- загруженные модули должны быть представлены пользователю в пользовательском интерфейсе с описанием модулей.

3. Приложение должно поддерживать загрузку соединений баз данных.

- Соединения базы данных хранятся в отдельном файле, организованном в формате .XML, хранящий строки подключения, названия и описание соединений;
- при запуске модуля необходимо поддерживать выбор нескольких соединений из всех возможных посредством пользовательского интерфейса.

4. Приложение должно осуществлять поддержку функций по построению модели базы данных.

- Модель базы данных заключается в формализованном внутреннем представлении архитектурных элементов реляционной базы данных. Приложение должно осуществлять анализ базы данных и построение данных моделей, описывающих структуру.

5. Приложение должно поддерживать конвертацию базы данных из MS SQL Server и Oracle в MS Access.

- Должен поддерживаться перенос элементов: главных и внешних ключей, таблиц, представлений;
- перенос таблиц, главных ключей и представлений осуществляется из указанной базы данных, перенос внешних ключей осуществляется из постоянной базы данных примерной структуры;
- статус каждого из этапов конвертации должен отображаться в пользовательском интерфейсе;
- при возникновении ошибки текст ошибки должен быть выведен;
- приложение должно поддерживать конвертацию нескольких баз данных одновременно;
- база данных под управлением СУБД MS Access должна быть скопирована.

3.2 Требования к составу и параметрам технических средств

Для функционирования программы необходим персональный компьютер со следующими минимальными требованиями к составу и параметрам технических средств:

- процессор Intel Pentium или совместимый,
- объем свободной оперативной памяти 2 Гб,
- объем необходимой памяти на жестком диске 10 Гб,
- стандартный VGA монитор,
- стандартная клавиатура.
- манипулятор «мышь».
- сеть, соединяющая устройство с данным приложением и устройство с базой данных, в которую вносится изменения скоростью более 10 Мбит/с.

3.3 Требования к информационной и программной совместимости

Программная система должна функционировать под операционной системой Microsoft Windows 7 или совместимой.

Язык пользовательского интерфейса программы — русский.

4 Требования к программной документации

В комплект документации должен входить документ «Руководство пользователя».

5 Технико-экономические показатели

Планируется распространять программу как freeware-продукт под GNU public licence.

6 Стадии и этапы разработки

Стадия	Результат	Время выполнения
Анализ предметной области	Отчёт по выбранных инструментах разработки, методологии разработки, стандарт оформления кода.	05.02.16 — 10.03.16
Проектирование и программная реализация	Исходные тексты, исполняемый файл, диаграммы архитектуры.	11.03.16 — 25.05.16
Документирование	Руководство пользователя	20.05.16 — 25.05.16

7 Порядок контроля и приемки

Приемка должна осуществляться в соответствии с документом «Техническое задание».

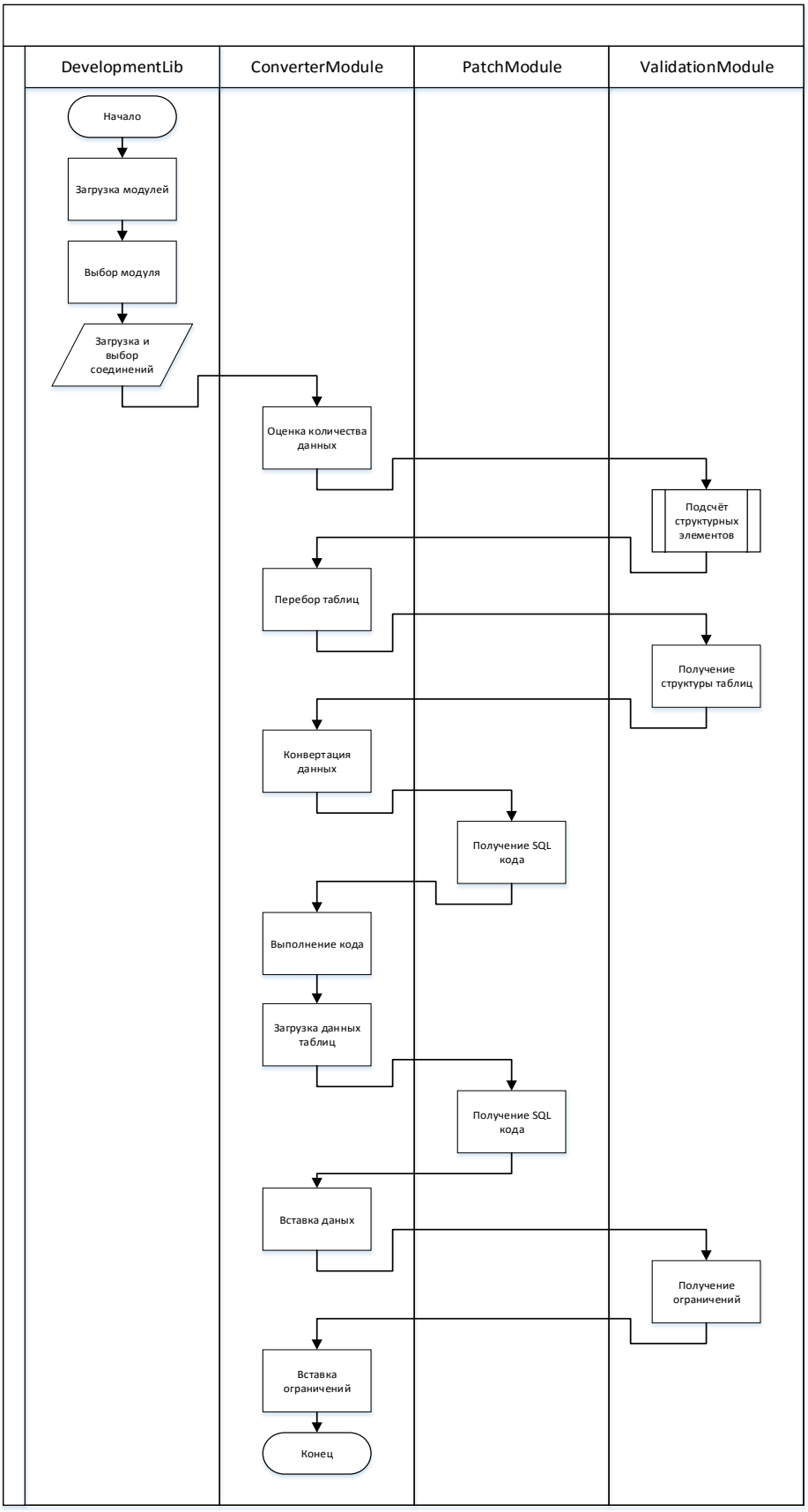
Приложение Б

Таблица сопоставления типов данных различных СУБД

	Integer	Floating point	Decimal	String	Binary	Date/Time	Boolean	Other
Microsoft SQL Server	TINYINT, SMALLINT, INT, BIGINT	FLOAT, REAL	NUMERIC, DECIMAL, SMALLMONEY, MONEY	CHAR, VARCHAR, TEXT, NCHAR, NVARCHAR, NTEXT	BINARY, VARBINARY, IMAGE, FILESTREAM	DATE, DATETIMEOFFSET, DATETIME2, SMALLDATETIME, DATETIME, TIME	BIT	CURSOR, TIMESTAMP, HIERARCHYID, UNIQUEIDENTIFIER, SQL_VARIANT, XML, TABLE, Geometry, Geography
Oracle	NUMBER	BINARY_FLOAT, BINARY_DOUBLE	NUMBER	CHAR, VARCHAR2, CLOB, NCLOB, NVARCHAR2, NCHAR, LONG (deprecated)	BLOB, RAW, LONG RAW, BFILE	DATE, TIMESTAMP (with/without TIMEZONE), INTERVAL	N/A	SPATIAL, IMAGE, AUDIO, VIDEO, DICOM, XMLType
Microsoft	LONG	DOUBLE	NUMERIC	TEXT, MEMO	BINARY, OLE	DATETIME	BIT	

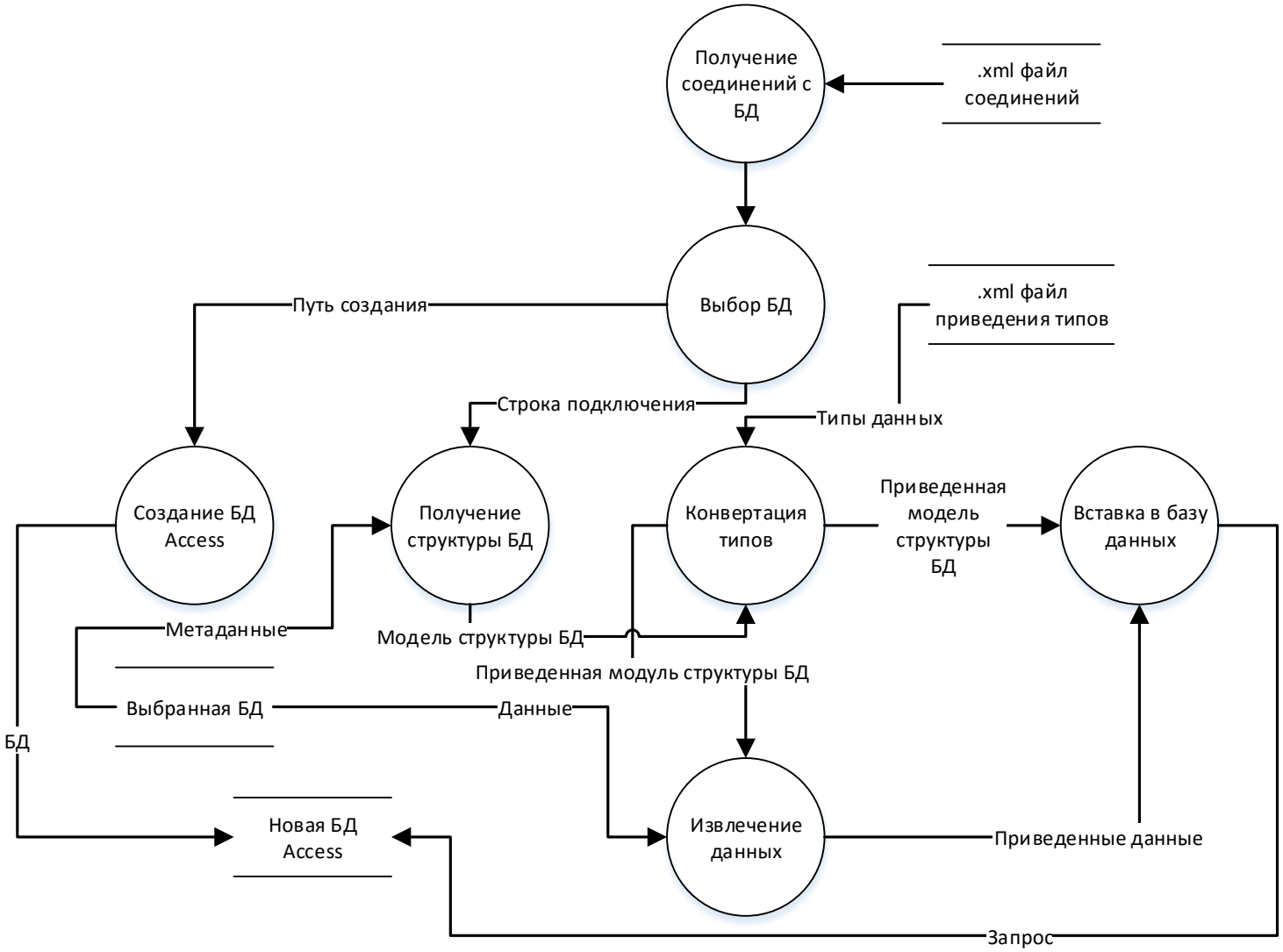
Приложение В

Схема алгоритма приложения



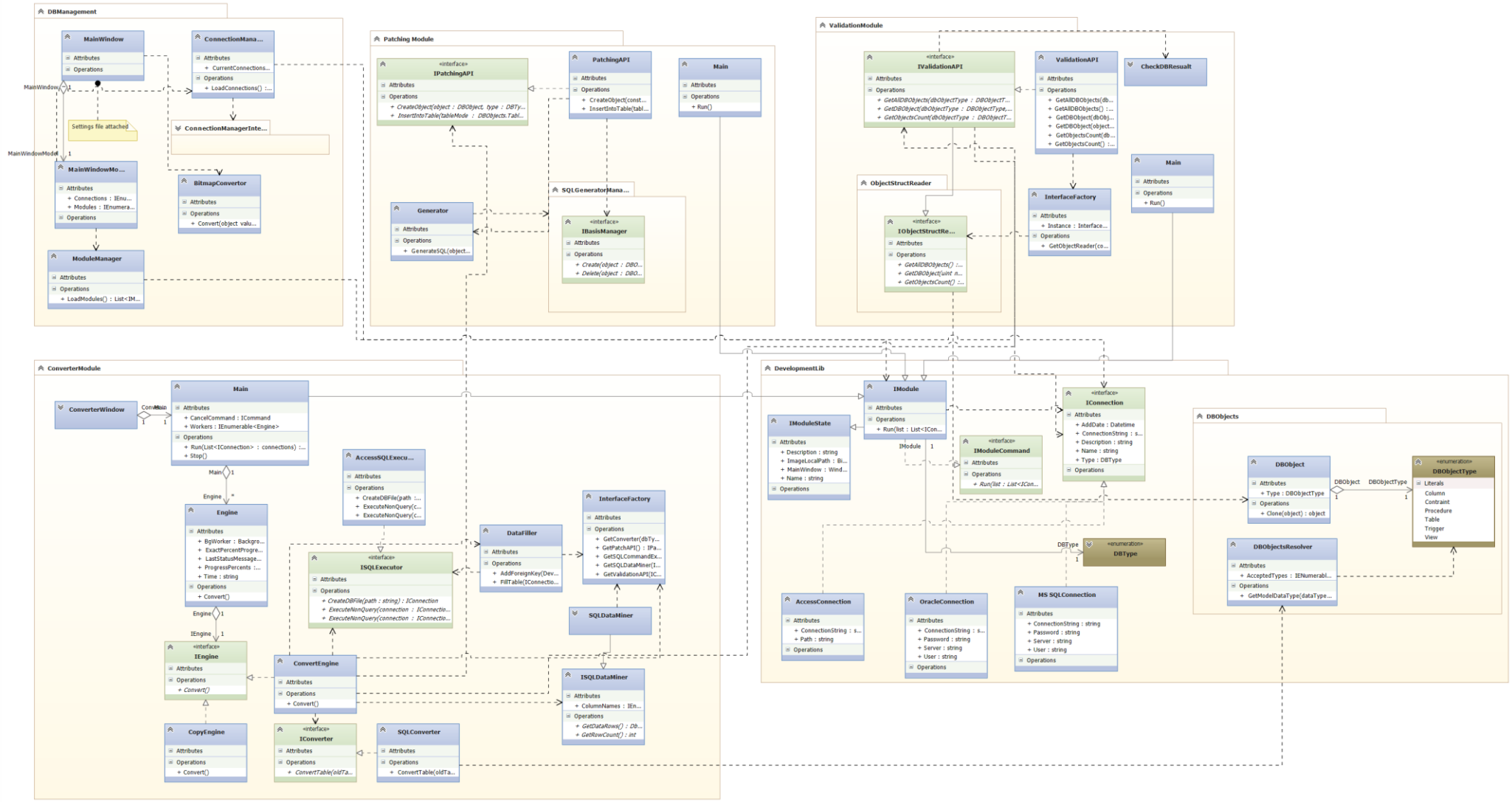
Приложение Г

Диаграмма потоков данных приложения в нотации Йордана/Де Марко



Приложение Д

Диаграмма классов UML приложения



Приложение Е

Календарный план-график выполнения проекта в виде диаграммы Ганта

№ работ	Вид работ	Исполнители	Tki, календарные	Продолжительность выполнения работ												
				февраль			март			апрель			май			июнь
				1	2	3	1	2	3	1	2	3	1	2	3	1
1	Создание ТЗ	Студент-программист	10													
1	Создание ТЗ	Руководитель проекта	2													
2	Изучение материалов	Студент-программист	12													
3	Выбор методологии	Руководитель проекта	2													
3	Выбор методологии	Студент-программист	6													
4	Составление плана работ	Руководитель проекта	1													
5	Выбор средств разработки	Руководитель проекта	1													
5	Выбор средств разработки	Студент-программист	8													
6	Выбор оформления кода	Руководитель проекта	1													
6	Выбор оформления кода	Студент-программист	3													
7	Паттерны программирования	Руководитель проекта	1													
7	Паттерны программирования	Студент-программист	3													
8	Средства подключения	Студент-программист	3													
9	Оценка технологий	Студент-программист	1													
10	Оценка целесообразности	Студент-программист	1													
11	Технологии предприятия	Студент-программист	4													
12	Разработка архитектуры	Руководитель проекта	3													
12	Разработка архитектуры	Студент-программист	40													
13	Эффективность архитектуры	Студент-программист	13													
14	Разработка интерфейса	Студент-программист	8													
15	Создание кода приложения	Студент-программист	17													
16	Текстирование	Студент-программист	15													
17	Написание документации	Студент-программист	14													
18	Создание ПЗ	Студент-программист	25													

- Студент-программист

- Руководитель проекта

Приложение Ж

Руководство пользователя

1 Введение

1.1 Назначение системы

Приложение позволяет осуществлять поддержку множества удаленных баз данных под управлением 3 типов СУБД: MS SQL, Oracle и MS Access. Путем реализации приложения производиться перенос данных в СУБД MS Access для уменьшения требуемых ресурсов СУБД, автоматизация деятельности по поддержке баз данных для экономии трудовых ресурсов.

1.2 Основные возможности

В программе реализованы возможности:

- загрузки и запуска модулей;
- загрузки соединений с базами данных и их выбор;
- осуществление конвертации базы данных.

1.3 Документация системы

Приложение поставляется в комплекте с электронным руководством пользователя.

2 Установка и удаление приложения

2.1 Требования к аппаратным и программным средствам

Для функционирования программы применяются следующие требования к персональному компьютеру:

- процессор Intel Pentium или совместимый,
- объем свободной оперативной памяти 2 Гб,
- объем необходимой памяти на жестком диске 10 Гб,
- стандартный VGA монитор,
- стандартная клавиатура.
- манипулятор «мышь».

- сеть, соединяющая устройство с данным приложением и устройство с базой данных, в которую вносится изменения скоростью более 10 Мбит/с.

2.2 Процедура установки

Установка приложения осуществляется путем копирования программных файлов в каталог на жестком диске.

2.3 Процедура удаления

Удаление приложения может быть произведено путем удаления его из каталога на жестком диске.

2.4 Запуск приложения

Запуск приложения осуществляется путем запуска исполняемого файла DBManagement.exe из директории программы на жестком диске.

3 Описание интерфейса

При запуске программы пользователь видит на экране форму (рисунок 1), где:

1. Кнопка запуска модуля конвертации.
2. Кнопки остальных не реализованных модулей.

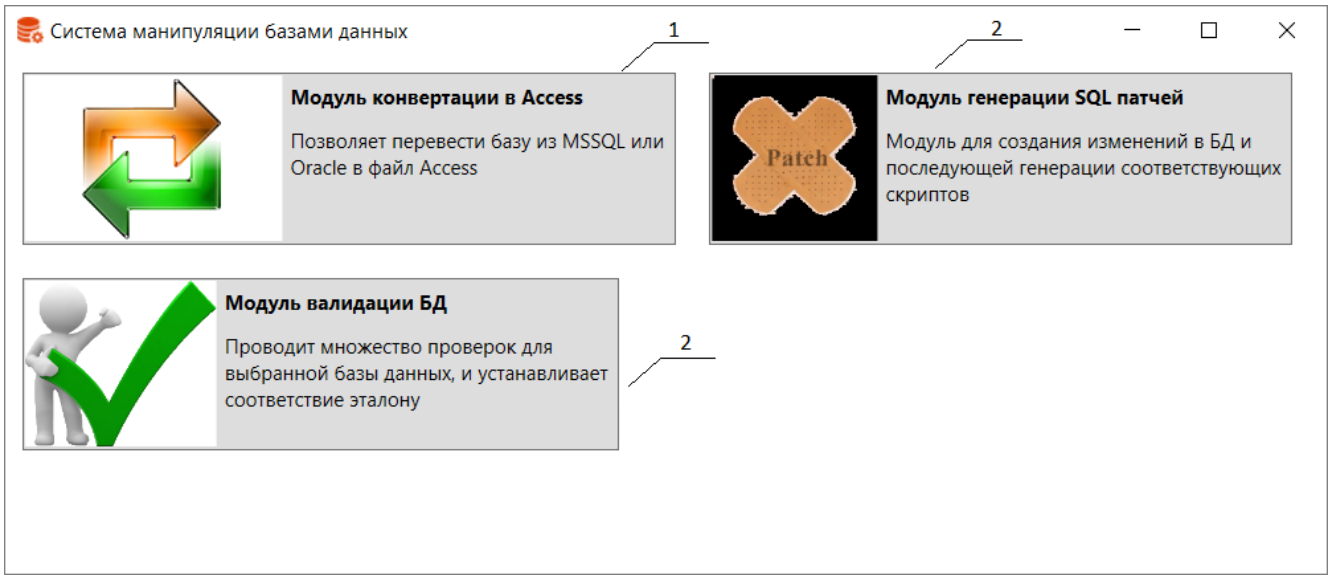


Рисунок 1 – Главное окно приложения

При выборе любого из модулей открывается окно выбора баз данных (рисунок 2), где:

1. Кнопка вывода подключений баз данных под управлением только MS SQL Server.
2. Кнопка вывода подключений баз данных под управлением только MS Access.
3. Кнопка вывода подключений баз данных под управлением только Oracle.
4. Кнопка вывода подключений всех баз данных.
5. Кнопки выбора баз данных.
6. Кнопка подтверждения выбранных баз данных.
7. Кнопка отмены выбора баз данных.

8. Поля, описывающие подключения баз данных.

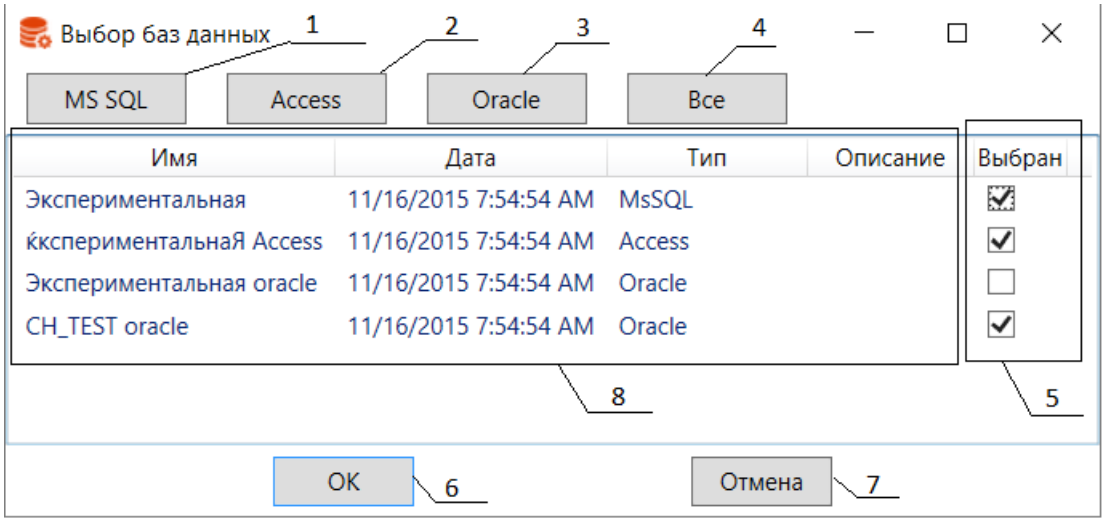


Рисунок 2 – Окно выбора баз данных

При выборе модуля конвертации баз данных после нажатия кнопки «ОК» выбора баз данных открывается окно конвертации баз данных (рисунок 3), где:

1. Поля, описывающие подключения баз данных.
2. Поля, описывающие статус конвертации баз данных.
3. Кнопка прерывания конвертирования баз данных, по завершению конвертации кнопка закрытия окна.

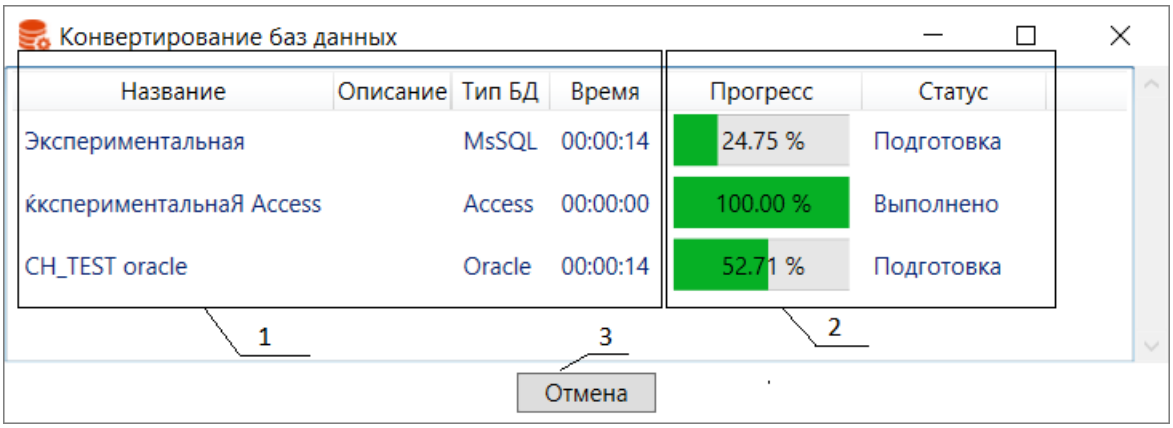


Рисунок 3 – Окно конвертации баз данных

4 Запуск конвертации баз данных

Для запуска конвертации баз данных требуется:

- запустить приложение;
- выбрать кнопку “Модуль конвертации в Access”;

- пометить в столбце “выбран” базы данных, которые необходимо конвертировать;
- нажать кнопку “ОК”;
- по завершению конвертации нажать кнопку “ОК”, которая переименовывается при завершении конвертации с кнопки “Отмена” или отменить её, нажав кнопку “Отмена” во время конвертации.