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

Школа: Инженерная школа информационных технологий и робототехники
Направление подготовки: 09.03.04 «Программная инженерия»
Отделение: Отделение информационных технологий

БАКАЛАВРСКАЯ РАБОТА

Тема работы
Алгоритмическое и программное обеспечение поиска и анализа данных социальных сетей

УДК 004.4.021:004.6:316.472.4

Студент

Группа	ФИО	Подпись	Дата
8К51	Осмоналиев Тимур Мирсланович		

Руководитель

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР ТПУ	Савельев Алексей Олегович	к.т.н.		

КОНСУЛЬТАНТЫ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОСГН ШБИП ТПУ	Подопригора Игнат Валерьевич	к.э.н.		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ООД ШБИП ТПУ	Винокурова Галина Федоровна	к.т.н.		

ДОПУСТИТЬ К ЗАЩИТЕ:

Руководитель ООП	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР ТПУ	Чердынцев Евгений Сергеевич	к.т.н.		

Томск – 2019 г.

ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ ПО ООП

Код результата	Результат обучения (выпускник должен быть готов)
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.
Р7	Использовать базовые и специальные знания в области проектного менеджмента для ведения комплексной инженерной деятельности.
Р8	Владеть иностранным языком на уровне, позволяющем работать в иноязычной среде, разрабатывать документацию, презентовать и защищать результаты комплексной инженерной деятельности.
Р9	Эффективно работать индивидуально и в качестве члена группы, состоящей из специалистов различных направлений и квалификаций, демонстрировать ответственность за результаты работы и готовность следовать корпоративной культуре организации.
Р10	Демонстрировать знания правовых, социальных, экономических и культурных аспектов комплексной инженерной деятельности.
Р11	Демонстрировать способность к самостоятельной к самостоятельному обучению в течение всей жизни и непрерывному самосовершенствованию в инженерной профессии.

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

Школа: Инженерная школа информационных технологий и робототехники
Направление подготовки: 09.03.04 «Программная инженерия»
Отделение школы: Отделение информационных технологий

УТВЕРЖДАЮ:
Руководитель ООП
_____Чердынцев Е.С.
(Подпись) (Дата)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

бакалаврской работы

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

Студенту:

Группа	ФИО
8К51	Осмоналиеву Тимуру Мирслановичу

Тема работы:

Алгоритмическое и программное обеспечение поиска и анализа данных социальных сетей	
Утверждена приказом директора (дата, номер)	14.05.2019 г. №3733/с

Срок сдачи студентом выполненной работы:	19.06.2019

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе *(наименование объекта исследования или проектирования; производительность или нагрузка; режим работы (непрерывный, периодический, циклический и т. д.); вид сырья или материал изделия; требования к продукту, изделию или процессу; особые требования к особенностям функционирования (эксплуатации) объекта или изделия в плане безопасности эксплуатации, влияния на окружающую среду, энергозатратам; экономический анализ и т. д.).*	Объектом исследования в данной работе является API и данные социальной сети «Вконтакте», рассматриваемым бизнес-процессом является сбор данных, для автоматизации которого необходимо разработать программную систему. Технологической платформой для разработки является Delphi Community Edition 10.3.

Перечень подлежащих исследованию, проектированию и разработке вопросов *(аналитический обзор по литературным источникам с целью выяснения достижений мировой науки техники в рассматриваемой области; постановка задачи исследования, проектирования, конструирования; содержание процедуры исследования, проектирования, конструирования; обсуждение результатов выполненной работы; наименование дополнительных разделов, подлежащих разработке; заключение по работе).*	1. Исследование предметной области 2. Проектирование и реализация программной системы 3. Тестирование работы разработанной программной системы 4. Финансовый менеджмент, ресурсоэффективность и ресурсосбережение 5. Социальная ответственность
Перечень графического материала *(с точным указанием обязательных чертежей)*	1. Диаграмма организационной структуры компании 2. Диаграммы в нотации IDEF0 3. Диаграмма вариантов использования в нотации UML 4. Диаграммы потоков данных в нотации DFD 5. Диаграмма классов в нотации UML 6. Скриншоты работы разработанной программной системы 7. Матрица SWOT 8. Таблица трудозатрат на выполнение проекта 9. Линейный график работ 10. Презентация Microsoft PowerPoint

Консультанты по разделам выпускной квалификационной работы
(с указанием разделов)

Раздел	Консультант
Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	Подопригора Игнат Валерьевич
Социальная ответственность	Винокурова Галина Федоровна

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	10.09.2018

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР ТПУ	Савельев Алексей Олегович	к.т.н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8К51	Осмоналиев Тимур Мирсланович		

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

Школа: Инженерная школа информационных технологий и робототехники

Направление подготовки: Программная инженерия

Уровень образования: Бакалавр

Отделение школы: Отделение информационных технологий

Период выполнения: осенний / весенний семестр 2018/2019 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:	19.06.2019

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
19.06.2019	Раздел 1. Исследование предметной области	20
19.06.2019	Раздел 2. Проектирование и реализация программной системы	20
19.06.2019	Раздел 3. Тестирование работы разработанной программной системы	20
19.06.2019	Раздел 4. Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	20
19.06.2019	Раздел 5. Социальная ответственность	20

Составил преподаватель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР ТПУ	Савельев Алексей Олегович	к.т.н.		

СОГЛАСОВАНО:

Руководитель ООП	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР ТПУ	Чердынцев Евгений Сергеевич	к.т.н.		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ»

Студенту:

Группа	ФИО
8К51	Осмоналиеву Тимуру Мирслановичу

Школа	ИШИТР	Кафедра	ОИТ
Уровень образования	Бакалавриат	Направление/специальность	09.03.04 Программная инженерия

Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:	
1. Стоимость ресурсов научного исследования (НИ): материально-технических, энергетических, финансовых, информационных и человеческих	Амортизационные отчисления – 4949,20 рублей, затраты на основную заработную плату 166480,55 рублей, затраты на отчисление во внебюджетные фонды – 46614,15 рублей, накладные расходы – 35157,52 рублей.
2. Нормы и нормативы расходования ресурсов	
3. Используемая система налогообложения, ставки налогов, отчислений, дисконтирования и кредитования	
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Оценка коммерческого потенциала, перспективности и альтернатив проведения НИ с позиции ресурсоэффективности и ресурсосбережения	Потенциальные потребители результатов исследования. Анализ конкурентных технических решений. SWOT – анализ.
2. Планирование научно-исследовательских работ	Структура работ в рамках научного исследования. Определение трудоемкости выполнения работ. Разработка графика проведения научного исследования. Бюджет научно-технического исследования.
3. Определение ресурсной (ресурсосберегающей), финансовой, бюджетной, социальной и экономической эффективности исследования	Определение интегрального финансового показателя разработки. Определение интегрального показателя ресурсоэффективности разработки. Определение интегрального показателя эффективности.
Перечень графического материала *(с точным указанием обязательных чертежей):*	
1. Оценка конкурентоспособности технических решений. 2. Матрица SWOT. 3. График проведения и бюджет НИ. 4. Оценка ресурсной, финансовой и экономической эффективности НИ.	
Дата выдачи задания для раздела по линейному графику	01.03.2019

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОСГН ШБИП ТПУ	Подопригора Игнат Валерьевич	к.э.н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8К51	Осмоналиев Тимур Мирсланович		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА «СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Студенту:

Группа	ФИО
8К51	Осмоналиеву Тимуру Мирслановичу

Школа	ИШИТР	Отделение (НОЦ)	ОИТ
Уровень образования	Бакалавриат	Направление/специальность	09.03.04 Программная инженерия

Тема ВКР:

Разработка коррекционно-развивающего приложения для умственно-отсталых детей младшего школьного возраста	
Исходные данные к разделу «Социальная ответственность»:	
1. Характеристика объекта исследования (вещество, материал, прибор, алгоритм, методика, рабочая зона) и области его применения	Объект исследования – коррекционно-развивающее приложение для умственно-отсталых детей Рабочее место – рабочий стол с персональным компьютером в общем помещении
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Правовые и организационные вопросы обеспечения безопасности: – специальные (характерные при эксплуатации объекта исследования, проектируемой рабочей зоны) правовые нормы трудового законодательства; – организационные мероприятия при компоновке рабочей зоны.	• Рабочее место при выполнении работ сидя регулируется ГОСТом 12.2.032 –78 • Организация рабочих мест с электронно-вычислительными машинами регулируется СанПиНом 2.2.2/2.4.1340 – 03 • Рациональная организация труда в течение рабочего времени предусмотрена Трудовым Кодексом РФ ФЗ-197
2. Производственная безопасность: 2.1. Анализ выявленных вредных и опасных факторов 2.2. Обоснование мероприятий по снижению воздействия	2.1 • Повышенный уровень электромагнитных излучений • Отклонение показателей микроклимата • Недостаточная освещенность рабочей зоны • Повышенный уровень шума • Повышенное значение напряжения в электрической цепи, замыкание которой может произойти через тело человека
3. Экологическая безопасность:	Анализ негативного воздействия на окружающую природную среду: утилизация компьютеров и другой оргтехники, использованных люминесцентных ламп, мусорных отходов, в том числе бумагу.
4. Безопасность в чрезвычайных ситуациях:	Возможные чрезвычайные ситуации: • Пожар

Дата выдачи задания для раздела по линейному графику	01.03.2019

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ООД ШБИП ТПУ	Винокурова Галина Федоровна	к.т.н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8К51	Осмоналиев Тимур Мирсланович		

Реферат

Выпускная квалификационная работа содержит 137 страниц, 22 рисунка, 13 таблиц, 6 приложений и 30 литературных источников.

Ключевые слова: социальные сети, сбор данных, программная автоматизация, API социальных сетей, социология.

Цель работы: разработка программной системы для поиска и сбора данных социальных сетей, которое позволит собирать неперсональную и доступную для получения информацию о пользователях, и хранить её в базе данных.

В процессе выполнения работы использовалась интегрированная среда разработки Delphi Community Edition 10.3, для создания программной системы.

В результате работы были разработана и протестирована программная система для сбора информации о пользователях социальной сети по API сервиса «Вконтакте».

В первой главе представлено описание предметной области, обзор и аналогов, описание выбранного средства разработки и описание разрабатываемой программной системы.

Во второй главе представлено моделирование, проектирование и разработка программной системы.

В третьей главе продемонстрирован процесс тестирования разработанной программной системы.

Четвертая глава представляет собой выполненное задание по разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение», при выполнении которого были использованы данные анализа в области проектного и финансового менеджмента, в том числе менеджмента рисков.

Пятая глава представляет собой выполненное задание по разделу «Социальная ответственность», где были рассмотрены аспекты производственной и экологической безопасности, безопасности в чрезвычайных ситуациях, а также правовые вопросы организации труда.

Список терминов и сокращений

1. IDE (Integrated development environment) – интегрированная среда для проектирования, разработки, компиляции и тестирования программного обеспечения.
2. МБ (мегабайт) – кратная единица измерения количества информации, равная 2^{20} = 1 048 576 байт.
3. ID (identifier) – уникальный код, позволяющий однозначно идентифицировать объект.
4. ПК – персональный компьютер.
5. ПО – программное обеспечение.
6. API (Application Programming Interface) – набор готовых программных объектов, которые реализуют взаимодействие между различными программными системами.
7. DFD (Data Flow Diagrams) – нотация, описывающая внешние источники данных для системы, логические функции, потоки данных и хранилища данных, к которым осуществляется доступ.
8. IDEF0 (Integration Definition for Function Modeling) – нотация, описывающая бизнес-процессы.
9. UML (Unified Modeling Language) – язык моделирования бизнес-процессов.

Оглавление

Введение

В современном мире социальные сети являются неотъемлемой частью жизни большинства людей. В социальных сетях люди публикуют о себе различную информацию. Сбор и анализ этой информации является мощным инструментом для социологических и маркетинговых исследований.

В настоящее время, для анализа аудитории социальных сетей и социальных групп используются преимущественно ручные методы сбора и обработки информации. Разрабатываемая программная система направлена на оптимизацию данного процесса.

Актуальность разработки программной системы по сбору пользовательской информации из социальных сетей состоит в том, что на текущий момент времени на рынке программного обеспечения отсутствуют легальные и доступные решения по данной области.

Целью разработки является проектирование и создание программной системы для поиска и сбора данных социальных сетей, которое позволит собирать неперсональную и доступную для получения информацию о пользователях, и хранить её в базе данных.

Для выполнения цели выдвинуты следующие задания:

1. Привести описание предметной области, включая объект автоматизации и основной рассматриваемый бизнес-процесс (диаграммы в нотации IDEF0).
2. Выполнить обзор существующих аналогов и провести анализ результатов.
3. Выделить роли пользователей и описать варианты использования разрабатываемой программной системы (диаграмма в нотации UML).
4. Смоделировать бизнес-процесс после внедрения разработки (диаграмма в нотации IDEF3).
5. Спроектировать разрабатываемую программную систему (диаграммы в нотации DFD).

6. Изучить процесс взаимодействия с социальными сетями посредством запросов API.

7. Реализовать спроектированную программную систему.

8. Провести тестирование разработанной программной системы в условиях, близких к реальному использованию.

Дополнительно в данной работе были выполнены задания по разделам «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение» и «Социальная ответственность», позволяющие оценить необходимость проведения данной работы и реальность внедрения результатов работы в реальную деятельность организаций.

Глава 1. Исследование предметной области

1.1. Описание предметной области

1.1.1. Описание объекта автоматизации

Объектом автоматизации является сбор данных из социальной сети «Вконтакте». Собираемые данные и метод их сбора должны удовлетворять следующим требованиям, изложенным в пользовательском соглашении «Вконтакте» [1], а именно:

1. Запрещены сбор и обработка персональных данных пользователей;
2. Запрещено использование вне интерфейса Сайта без разрешения администрации;
3. Приложения обязаны использовать только разработанные «Вконтакте» методы API, ID и пользовательский ключ доступа, указанные в настройках данных приложений. Использование других методов API не разрешается.

Рассмотрим информацию, собираемую методами API «Вконтакте» [2]:

1. Идентификатор пользователя;
2. Имя пользователя;
3. Фамилия пользователя;
4. Удалена или заблокирована ли страница;
5. Скрыта ли страница;
6. Поле «О себе»;
7. Поле «Деятельность»;
8. Дата рождения;
9. Любимые книги;
10. Любимые фильмы;
11. Любимые игры;
12. Любимая музыка;
13. Любимые телешоу;
14. Любимые сериалы;

15. Карьера:
    - Наименование компании;
    - Город;
    - Год начала работы;
    - Год окончания работы;
    - Должность.
16. Профиль в Skype, Facebook, Instagram, Livejournal, Twitter;
17. Номер мобильного телефона;
18. Номер домашнего телефона;
19. Количество фотографий, аудиозаписей, видеозаписей, фотоальбомов, заметок, друзей, сообществ, друзей онлайн, подписчиков, количество интересных для пользователя страниц;
20. Фотография пользователя и её миниатюра;
21. Адрес страницы;
22. Образование:
    - Название университета;
    - Название факультета;
    - Год окончания.
23. Родной город;
24. Интересы;
25. Время последнего посещения;
26. Тип платформы, с которой заходил пользователь:
    - Мобильная версия сайта;
    - Приложение для Android;
    - Приложение для Iphone;
    - Приложение для Ipad;
    - Приложение для Windows 10;
    - Приложение для Windows Phone;
    - Десктопная (полная) версия сайта.
27. Девичья фамилия;

28. Военная служба:
    - Номер части;
    - Год начала службы;
    - Год окончания службы.
29. Отчество;
30. Находится ли пользователь онлайн;
31. Политические предпочтения;
32. Изученные языки;
33. Мировоззрение;
34. Источники вдохновения;
35. Главное в людях для пользователя;
36. Главное в жизни пользователя;
37. Отношение к курению;
38. Отношение к алкоголю;
39. Любимые цитаты;
40. Список родственников пользователя:
    - Сын или дочь;
    - Брат или сестра;
    - Отец или мать;
    - Дедушка или бабушка;
    - Внук или внучка.
41. Семейное положение:
    - Не женат или не замужем;
    - Есть друг или есть подруга;
    - Помолвлен или помолвлена;
    - Женат или замужем;
    - Всё сложно;
    - В активном поиске;
    - Влюблен или влюблена;
    - В гражданском браке;

- Не указано.

42. Школа;
    - Страна и город;
    - Наименование школы;
    - Год начала обучения;
    - Год окончания обучения;
    - Год выпуска;
    - Буква класса;
    - Специализация;
    - Тип учебного заведения.
43. Пол;
44. Статус пользователя;
45. Подтверждена ли страница;
46. Университет:
    - Страна и город;
    - Наименование университета;
    - Наименование факультета;
    - Наименование кафедры;
    - Год окончания обучения;
    - Форма обучения;
    - Статус (например, выпускник или специалист).

Собирать вышеуказанные данные можно с помощью запроса к методу API «Вконтакте» users.get [3]. Для получения списка адресов страниц пользователей, к которым программная система будет применять API-запрос users.get будет использоваться метод API groups.getMembers [4], применимый к группам социальной сети «Вконтакте». Данный запрос позволяет получить все адреса страниц пользователей, состоящих в какой-либо группе «Вконтакте».

Для соблюдения федерального закона РФ от 27 июля 2006 года №152-ФЗ «О персональных данных», собираемая информация должна быть неперсональной [5]. Это осуществляется посредством применения методологии

деперсонализации данных [6], а именно: среди всех возможных данных, предоставляемых API-запросом users.get, необходимо убрать данные, содержащие персональную информацию, т.е. информацию, позволяющую однозначно идентифицировать пользователя социальной сети.

Для обеспечения деперсонализации собираемой информации необходимо удалить из API-запроса users.get следующие параметры:

1. Идентификатор пользователя;
2. Имя пользователя;
3. Фамилия пользователя;
4. Отчество пользователя;
5. Профиль в Skype, Facebook, Instagram, Livejournal, Twitter;
6. Номер мобильного телефона;
7. Номер домашнего телефона;
8. Фотография пользователя и её миниатюра;
9. Девичья фамилия;
10. Список родственников пользователя.

1.1.2. Описание рассматриваемого бизнес-процесса

В данной работе рассматриваемым бизнес-процессом является сбор и хранение данных из социальной сети «Вконтакте». Далее представлено описание бизнес-процесса.

Предположим, что существует задача по сбору данных пользователей из группы Томского политехнического университета в социальной сети «Вконтакте», для проведения социологических исследований. Человеку, выполняющему задачу, необходимо:

1. Зайти в социальную сеть «Вконтакте», используя свой логин и пароль;
2. Открыть страницу исследуемой группы;
3. Открыть список участников исследуемой группы;

4. Поочередно открывать каждую страницу участников исследуемой группы и выписывать все неперсональные данные о каждом пользователе.

На рисунке 1 представлена диаграмма первого уровня в нотации IDEF0 [7], графически описывающая рассматриваемый бизнес-процесс.

Рисунок 1 - Диаграмма первого уровня в нотации IDEF0 («черный ящик») для бизнес-процесса «Сбор данных из социальной сети Вконтакте»

Вход – данные пользователей социальной сети.

Управление:

- Задание по сбору данных;
- Федеральной закон РФ от 27 июля 2006 года №152-ФЗ «О персональных данных» [5].

Механизмы:

- Оператор;
- Оборудование – персональный компьютер, браузер, механизмы доступа к сети Интернет.

Выход – заполненная база данных с неперсональной информацией о пользователях.

На рисунке 2 представлена диаграмма второго уровня, декомпозирующая диаграмму первого уровня для бизнес-процесса «Сбор данных из социальной сети Вконтакте».

Рисунок 2 - Диаграмма второго уровня в нотации IDEF0
для бизнес-процесса «Сбор данных из социальной сети Вконтакте»

Декомпозиция главного бизнес-процесса показывает, что работа с заказом клиента разбивается на три этапа: выбрать критерии для поиска (наименование группы и перечень данных, которые необходимо собрать для задачи), ручной сбор данных и запись данных в БД.

На рисунке 3 представлена диаграмма третьего уровня, декомпозирующая этап «Ручной сбор данных».

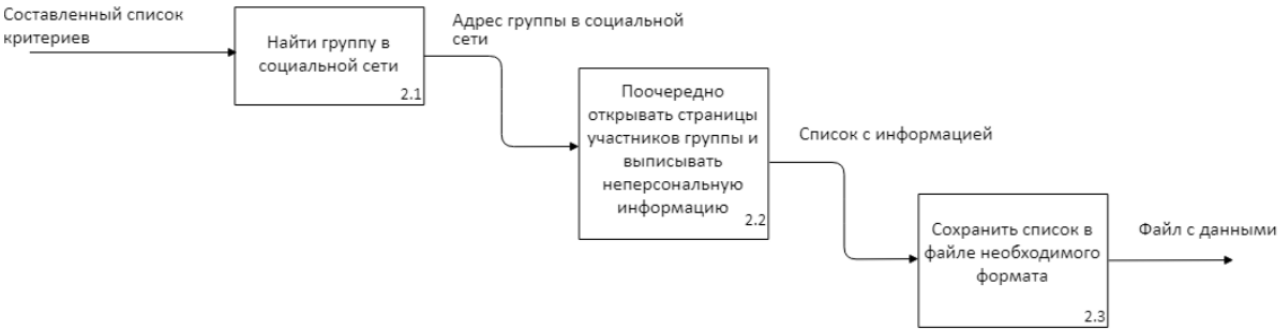


Рисунок 3 - Диаграмма третьего уровня в нотации IDEF0
для этапа «Ручной сбор данных» исследуемого бизнес-процесса

Оператору необходимо найти исследуемую группу в социальной сети «Вконтакте», открыть список пользователей, состоящих в группе, и путем

поочередного открытия страниц выписать всю неперсональную информацию, сохранив данные в файл необходимого формата.

Диаграмма Fish Bone [8], представленная на рисунке 4, используется для анализа слабых сторон проекта, значительную часть которых должна решить информационная система в результате ее внедрения в бизнес-процесс.

Рисунок 4 - Диаграмма Fish Bone для исследуемого бизнес-процесса

Как следует из диаграммы Fish Bone, исследуемый бизнес-процесс без внедрения информационной системы имеет ряд существенных недостатков, которые приводят к большим временным затратам на сбор, обработку и анализ данных.

1.2. Обзор существующих аналогов

В настоящее время существует небольшое количество программных систем и сервисов для сбора данных пользователей групп социальной сети «Вконтакте» в полуавтоматическом виде. Все существующие программные средства имеют различные преимущества и недостатки, но не до конца удовлетворяют покупателей данного программного обеспечения. По этой причине появляется необходимость в разработке нового решения.

1.2.1. «Agora Pulse»

Это программное обеспечение представляет из себя программу для сбора и анализа данных различных социальных сетей [9]. Программа является платной и запрещена для коммерческого пользования. Стоимость данного программного обеспечения составляет от 17449 до 33145 рублей в месяц за одну лицензию (на один персональный компьютер). Проанализировав данную программную систему, можно сделать вывод, что она имеет следующие характеристики:

- Возможность сохранять собранные данные в базу данных;
- Возможность работать с такими социальными сетями, как «Facebook, «Twitter», «Instagram», «LinkedIn», «YouTube»;
- Скорость выполнения сбора данных составляет 8 секунд за 1000 пользователей;
- Большое количество критериев для сбора данных;
- Ограниченное количество запросов в день (1000);
- Высокая стоимость лицензии;
- Запрещено коммерческое пользование;
- Отсутствует сбор данных из социальной сети «Вконтакте».

1.2.2. «YouScan»

Данный инструмент также является программой для сбора данных из социальных сетей [10]. Программа является платной и запрещена для коммерческого пользования. Стоимость данного программного обеспечения составляет от 77458 до 155022 рублей в месяц за одну лицензию (на один персональный компьютер). Проанализировав данную программную систему, можно сделать вывод, что она имеет следующие характеристики:

- Возможность сохранять собранные данные в форматы XLSX, PDF, PPTX;
- Возможность работать с такими социальными сетями, как «Facebook», «Twitter», «YouTube», «Вконтакте».
- Скорость выполнения сбора данных в 5 секунд за 1000 пользователей;
- Большое количество критериев для сбора данных;
- Неограниченное количество запросов в день;
- Высокая стоимость лицензии;
- Запрещено коммерческое пользование;
- Присутствует сбор данных из социальной сети «Вконтакте».

1.2.3. Результат обзора и анализа аналогов

По результатам анализа конкурентов разрабатываемого программного обеспечения были изучены достоинства программ-конкурентов и их недостатки. Некоторые из представленных конкурентов имеют широкий функционал, но не совпадают с требованиями российских пользователей. Наиболее популярной социальной сетью на территории Российской Федерации является социальная сеть «Вконтакте», и не все представленные программы-конкуренты способны осуществлять из неё сбор информации. Также, отличительными характеристиками конкурентов является высокая стоимость программного обеспечения. Следовательно, можно сделать следующий вывод:

разработка программной системы, рассматриваемой в данной работе, является актуальной.

1.3. Описание выбранного средства разработки

Для разработки программной системы был выбран язык Delphi [11], поскольку программные продукты на данном языке обладают высокой производительностью и низкими требованиями к ресурсам персонального компьютера пользователя. Одноименная среда разработки компьютерных программ располагает улучшенной отладкой программного обеспечения, имеет высокоскоростной компилятор (является самым быстрым компилятором в мире), оптимальным функционалом для облегченного создания пользовательского интерфейса.

1.4. Описание разрабатываемой программной системы

Данная работа предполагает разработку программной системы со следующими функциональными и техническими требованиями:

1. Выбор пользователем группы социальной сети «Вконтакте» для сбора информации;
2. Отсутствие ошибок при сборе данных из-за того, что соответствующая информация не была указана пользователем социальной сети;
3. Сбор доступных неперсональных пользовательских данных из социальной сети «Вконтакте»;
4. Запись собранной информации в базу данных MySQL и файл электронных таблиц Excel;
5. Проверка собираемой информации на факт деперсонализации.

Нефункциональные требования

1. Соответствие программы действующим законам и подзаконным актам РФ;

2. Отклик программы в ответ на запрос по сбору информации должен составлять не более 10 секунд;
3. Программа должна работать без сбоев и аварийных завершений работы;
4. Программа должна быть разработана на языке программирования Delphi в среде разработки Delphi Community Edition 10.3;
5. База данных должна быть разработана при помощи технологии MySQL;
6. Файл электронных таблиц должен быть разработан при помощи технологии Excel;
7. Программа должна поддерживать неограниченное количество запросов в день;
8. Стоимость программной системы должна быть ниже стоимости аналогов на рынке. Согласно анализу, цена альтернативных решений варьируется от 17449 рублей в месяц (минимальная цена для Agora Pulse) до 155022 рубля в месяц (максимальная цена для YouScan). Предполагаемая стоимость покупки пользователем разрабатываемой программной системы составляет 5000 рублей за одну единицу в месяц.
9. Взаимодействие программы с пользователем должно выполняться при помощи пользовательского интерфейса (реализация при помощи формы).

Глава 2. Проектирование и реализация программной системы

2.1. Роли пользователей в системе. UML-моделирование

Используемая система предусматривает только роль оператора. На рисунке 5 представлена диаграмма вариантов использования в нотации Unified Modeling Language (UML) [12]. На диаграмме изображены варианты использования, которые относятся к роли «Оператор».

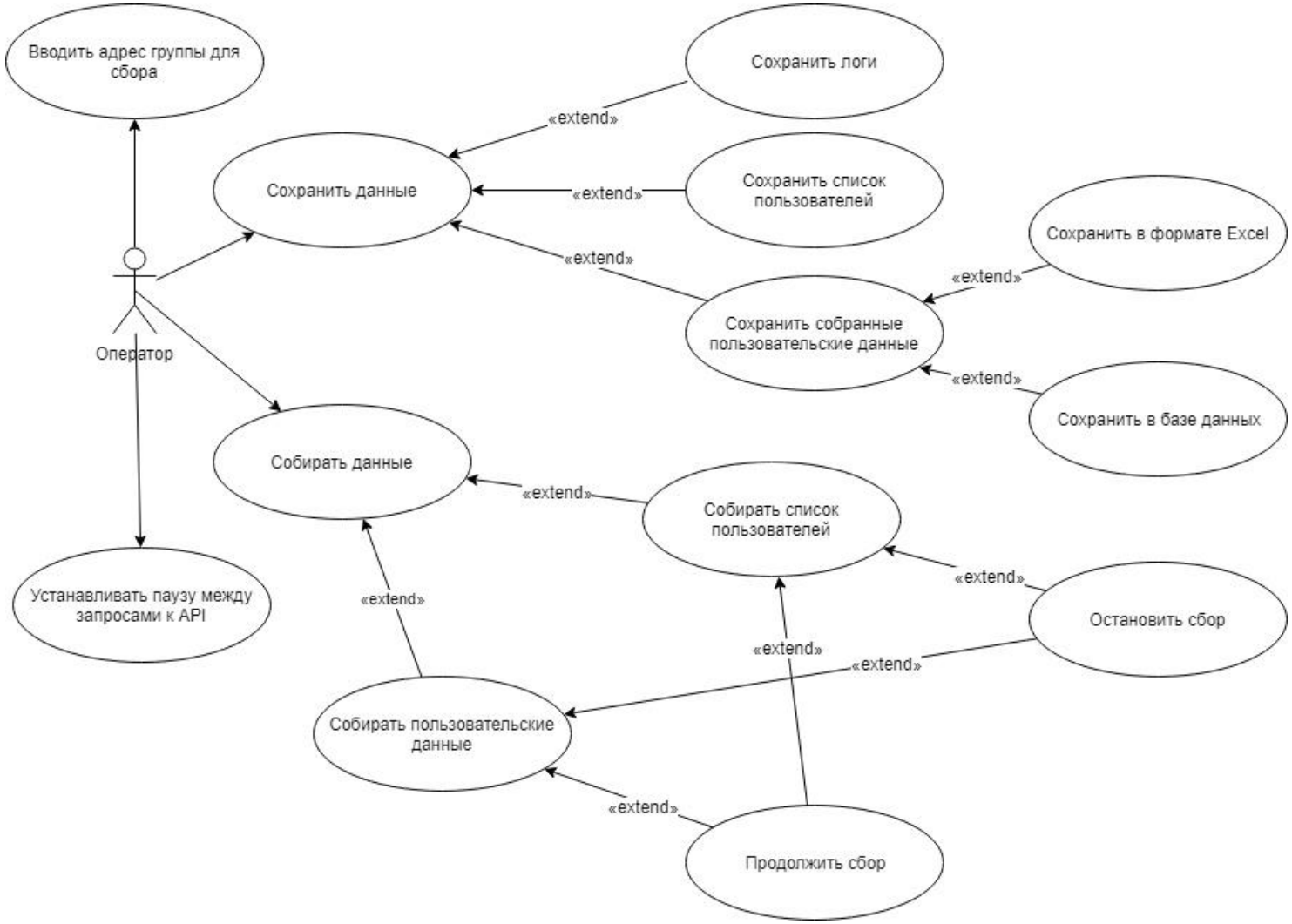


Рисунок 5 - Диаграмма вариантов использования для роли «Оператор»

2.2. Функциональное моделирование процесса

Задачей разрабатываемой программной системы является автоматизация процесса сбора неперсональных данных из социальной сети «Вконтакте».

Ниже продемонстрированы диаграммы IDEF0 для бизнес-процесса после внедрения разрабатываемой системы. На рисунке 6 представлена диаграмма первого уровня в нотации IDEF0, графически описывающая рассматриваемый бизнес-процесс после внедрения программной системы.

Рисунок 6 - Диаграмма первого уровня в нотации IDEF0 («черный ящик») для бизнес-процесса «Сбор данных из социальной сети Вконтакте»

Вход – данные пользователей социальной сети.

Управление:

- Задание по сбору данных;
- Федеральной закон РФ от 27 июля 2006 года №152-ФЗ «О персональных данных».

Механизмы:

- Оператор;
- Оборудование – персональный компьютер, браузер, механизмы доступа к сети Интернет.

Выход – заполненные база данных и файл Excel с неперсональной информацией о пользователях.

На рисунке 7 представлена диаграмма второго уровня, декомпозирующая диаграмму первого уровня для бизнес-процесса «Сбор данных из социальной сети Вконтакте».

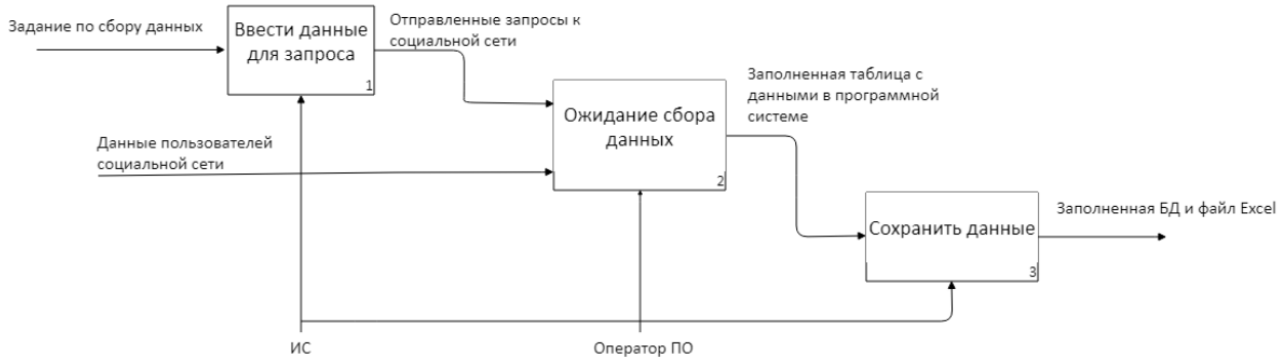


Рисунок 7 - Диаграмма второго уровня в нотации IDEF0
для бизнес-процесса «Сбор данных из социальной сети Вконтакте»

Декомпозиция главного бизнес-процесса показывает, что работа с заказом клиента разбивается на три этапа: ввести данные для запроса (адрес группы и API-токен), ожидание сбора данных и сохранение данных.

На рисунке 8 представлена диаграмма третьего уровня, декомпозирующая этап «Ввести данные для запроса».

Рисунок 8 - Диаграмма третьего уровня в нотации IDEF0
для этапа «Ввести данные для запроса» исследуемого бизнес-процесса

Оператору необходимо ввести в соответствующие текстовые поля адрес исследуемой группы, токен доступа и нажать на кнопку «Старт».

2.3. Моделирование потоков данных программной системы

Для описания потоков данных, которые будут реализованы с помощью программной системы, были разработаны диаграммы потоков данных с использованием методологии Data Flow Diagrams (DFD) [13]. Данная методология позволяет провести графический структурный анализ, описывая внешних по отношению к системе источников и адресатов, их взаимосвязь с системой, логические функции, потоки и хранилища данных.

На рисунке 9 представлена общая диаграмма потоков данных, включающая в себя все необходимые функциональные возможности программной системы.

Рисунок 9 - Общая диаграмма потоков данных в разрабатываемой программной системе в нотации DFD

На рисунке 10 представлена декомпозиция DFD.

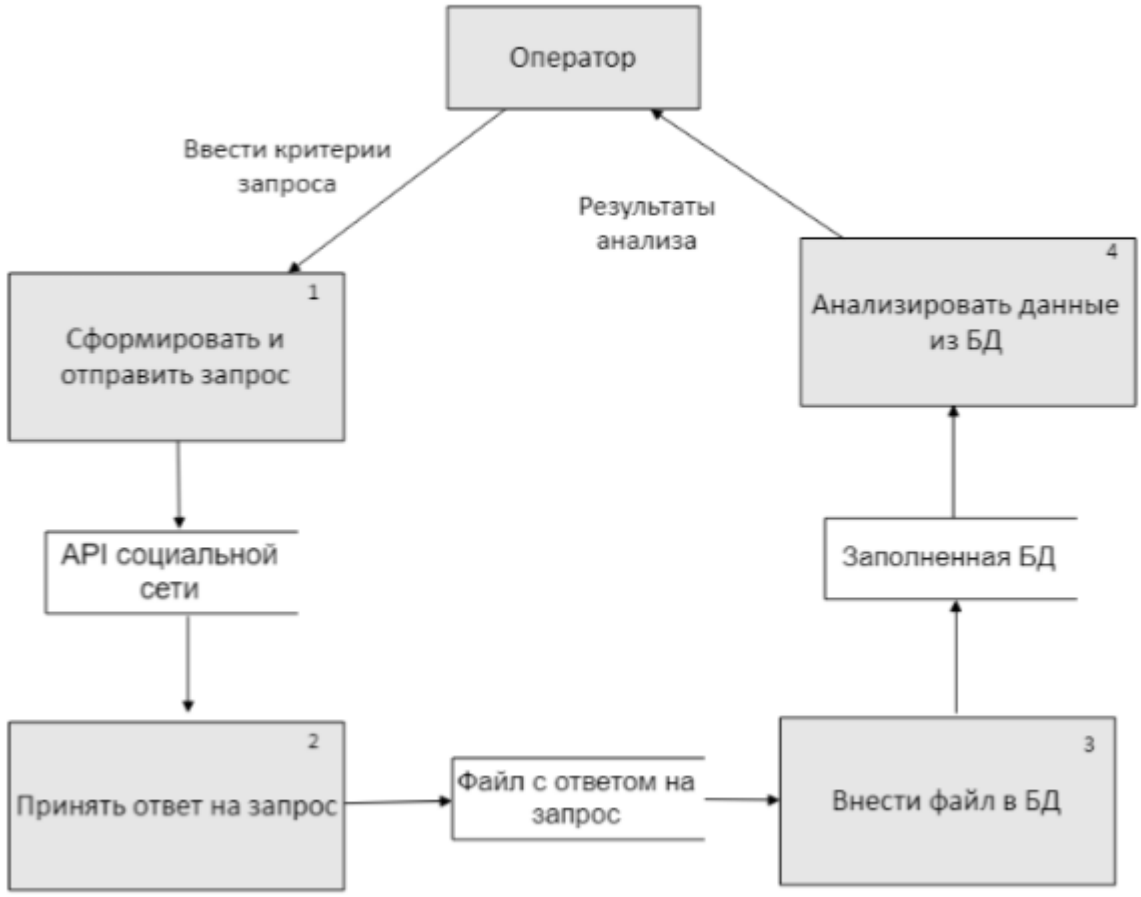


Рисунок 10 - Декомпозиция диаграммы потоков данных в нотации DFD

На рисунке 11 представлена декомпозиция DFD для этапа рассматриваемого процесса «Сформировать и отправить запрос».

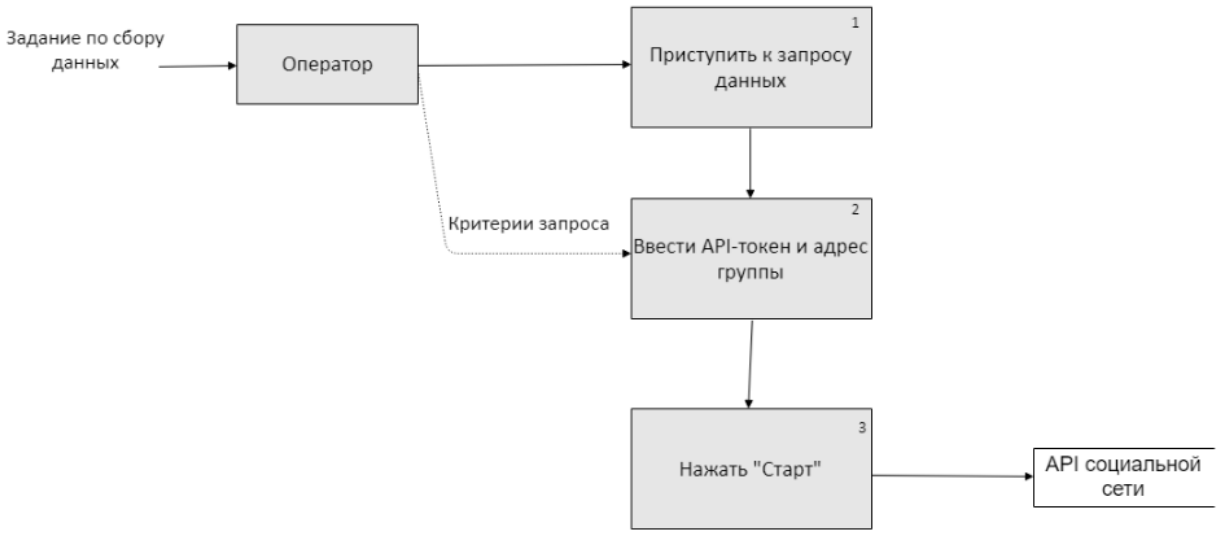


Рисунок 11 - Декомпозиция диаграммы потоков данных в нотации DFD для этапа «Сформировать и отправить запрос»

2.4. Описание объектов системы

Диаграммы EPC применяются для бизнес-моделирования, для улучшения и оптимизации бизнес-процесса [14]. Диаграмма EPC для исследуемого бизнес-процесса продемонстрирована на рисунке 12.

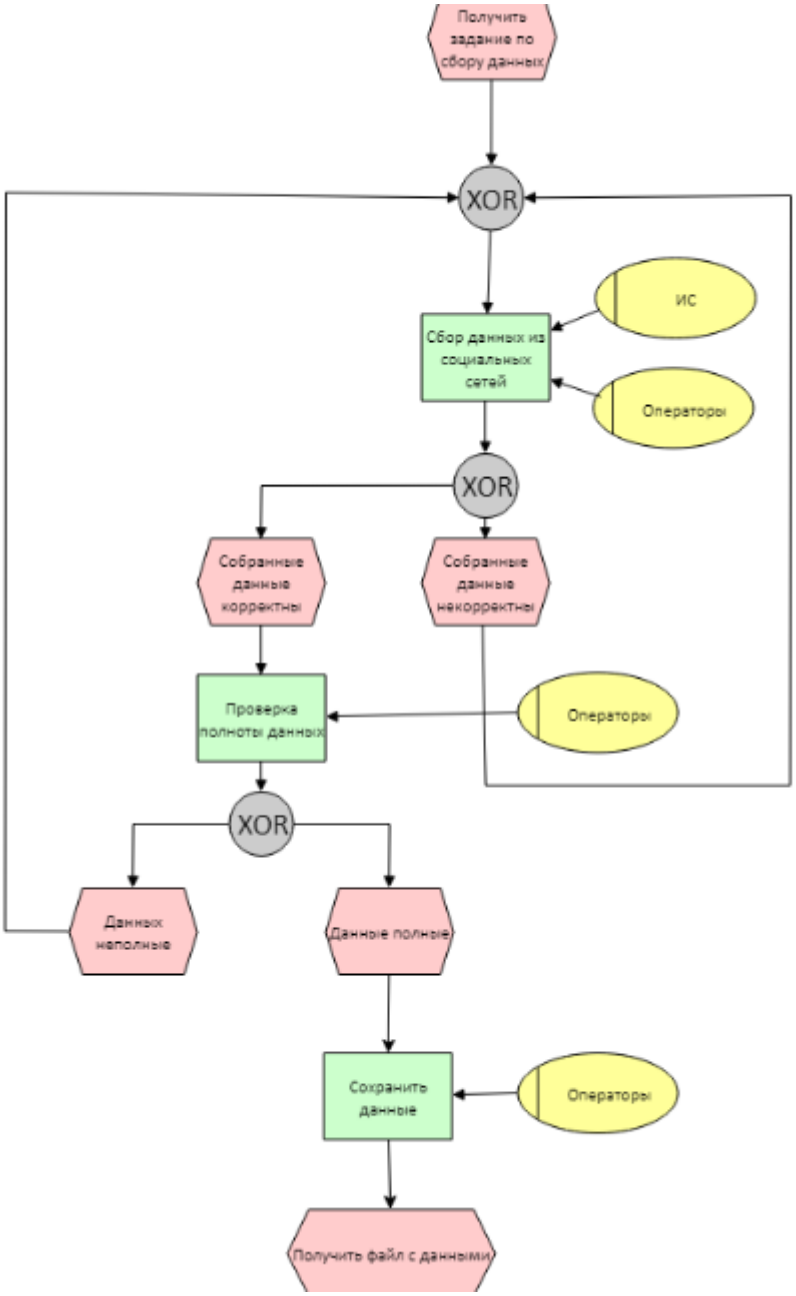


Рисунок 12 - Диаграмма EPC

Диаграмма BPMN [15] служит для описания модели бизнес-процесса. Для исследуемого бизнес-процесса продемонстрирована на рисунке 13.

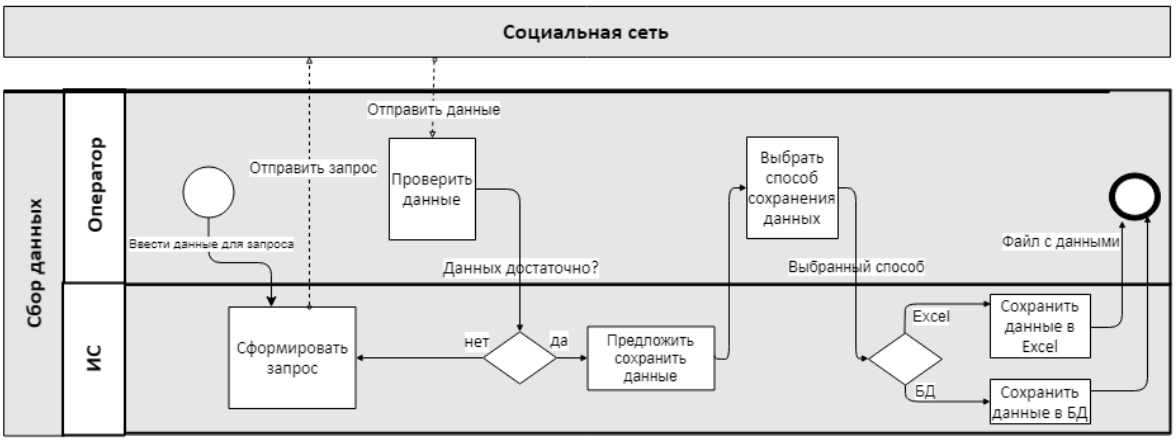


Рисунок 13 - Диаграмма BPMN

Ниже приведен список наименований пользовательских данных, подлежащих сбору программной системой:

1. deactivated – удалена ли страница пользователя;
2. is_closed – открыта ли страница для просмотра;
3. about – о себе;
4. activities – деятельность;
5. verified – подтверждена ли страница;
6. bdate – день рождения;
7. books – любимые книги;
8. career – данные о карьере;
9. city – наименование города проживания;
10. connections – есть ли профили в других социальных сетях;
11. country – наименование страны проживания;
12. education – данные об образовании;
13. followers_count – количество подписчиков;
14. games – любимые игры;
15. has_mobile – указан ли номер мобильного телефона;
16. has_photo – загружена ли главная фотография профиля;
17. home_town – наименование родного города;
18. interests – интересы;
19. last_seen – дата и время последнего посещения страницы пользователем;

20. movies – любимые фильмы;
21. music – любимая музыка;
22. occupation – данные о текущем роде занятий;
23. online – находится ли пользователь сейчас на сайте;
24. personal – данные о жизненной позиции пользователя;
25. quotes – любимые цитаты;
26. relation – семейное положение;
27. schools – данные о школе;
28. sex – пол пользователя;
29. site – веб-сайт;
30. status – текст статусной строки пользователя;
31. timezone – временная зона пользователя;
32. trending – есть ли метка популярности у пользователя;
33. tv – любимые телешоу;
34. universities – данные об учебе в университете.

Глава 3. Тестирование работы программной системы

3.1. Сбор страниц для анализа

С целью выявления ошибок и отладки разработанной программной системы в условиях, близких к реальному использованию, было проведено функциональное тестирование исследовательским методом.

Для тестирования была выбрана группа «ТПУ | Томский политехнический университет» [16], размещенная в социальной сети «Вконтакте». Количество участников в ней составляет более 20500 человек.

Методом создания приложения был получен API-токен доступа, позволяющий разработчикам взаимодействовать с запросами API, предоставляемой социальной сетью «Вконтакте». Он был введен в текстовое поле «API-ключ». В поле «Группа» был введен адрес группы «ТПУ | Томский политехнический университет», а именно: https://vk.com/tpunews. Для поля «Пауза между запросами» введено значение в 5 секунд, поскольку у запросов к API социальной сети «Вконтакте» существует лимит. После была нажата кнопка «Собрать» для запуска механизма сбора списка адресов всех пользователей, находящихся в исследуемой группе. Это представлено на рисунке 14.

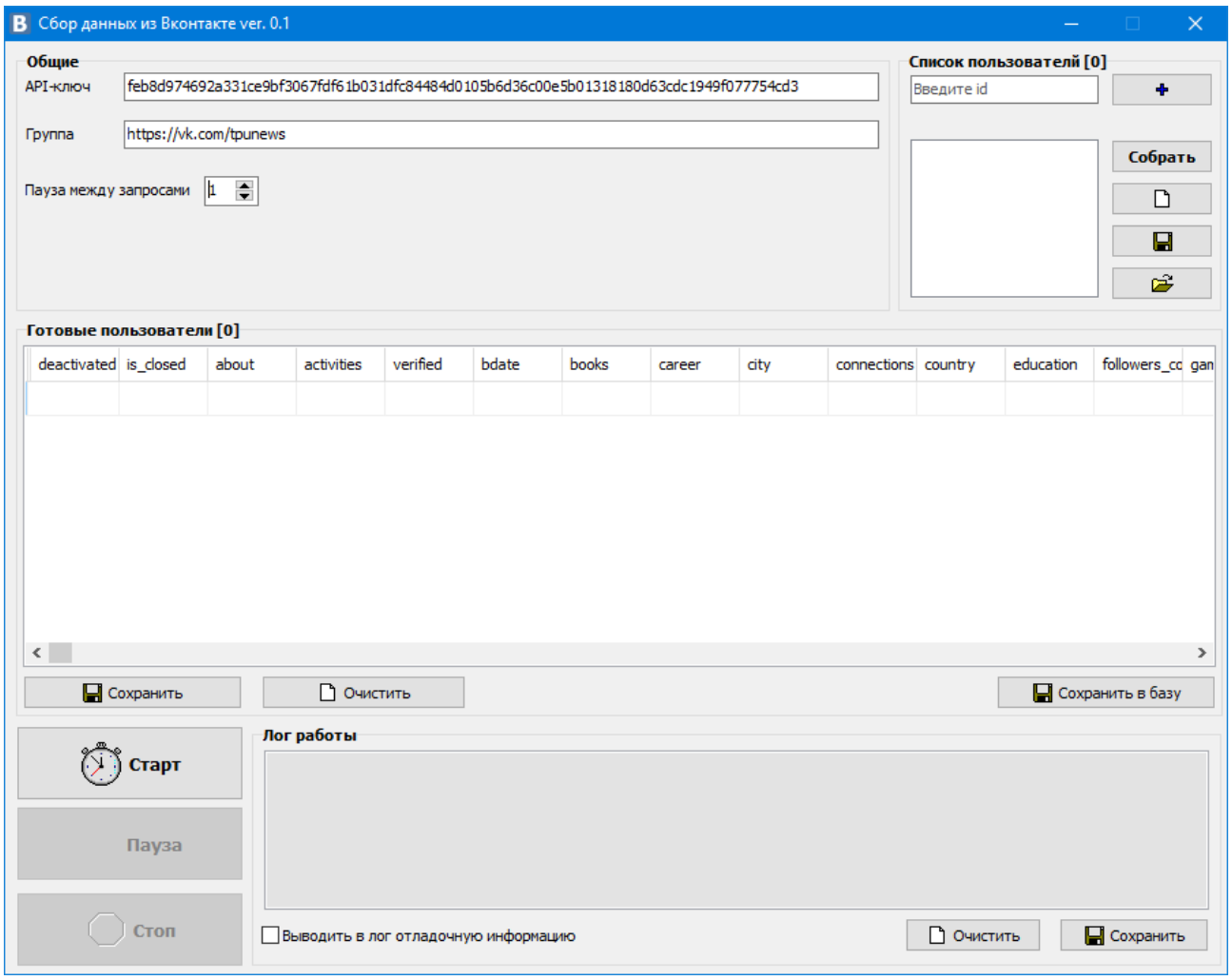


Рисунок 14 - Запуск механизма сбора адресов пользователей

Сбор информации о идентификаторах страниц всех 20539 участников исследуемой группы занял 72 секунды при выставлении соответствующего параметра «Пауза между запросами» на минимум (1 секунда). Рисунок 15 демонстрирует завершение механизма сбора страниц.

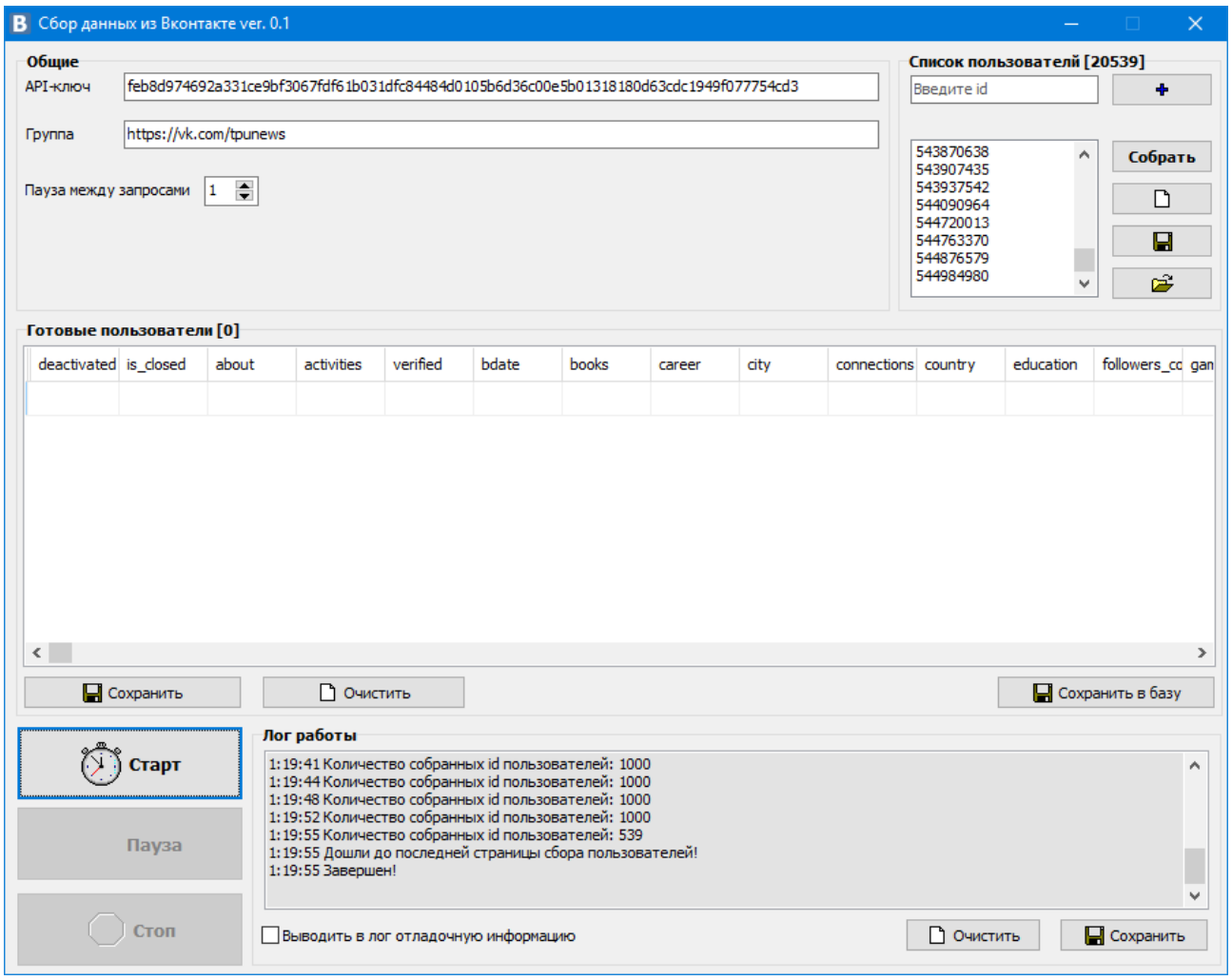


Рисунок 15 - Завершение работы механизма сбора адресов пользователей

По завершению работы механизма сбора адресов пользователей была нажата кнопка «Сохранить список адресов». Программа предлагает указать директорию для сохранения текстового файла формата .txt. Пример текстового файла продемонстрирован на рисунке 16.

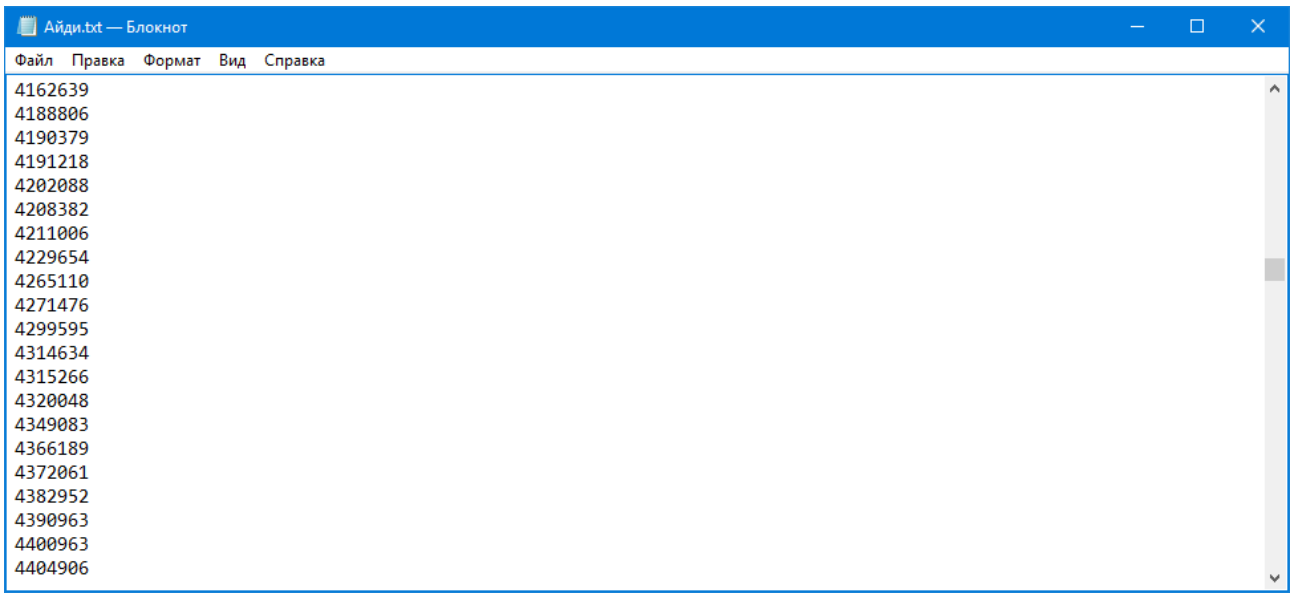


Рисунок 16 - Пример текстового файла для сохранения найденных в группе социальной сети «Вконтакте» адресов

Количество записей в файле равняется фактическому числу участников исследуемой группы и составляет 20539 записей.

3.2. Сбор данных

После выполнения действий из предыдущей главы, было произведено нажатие на кнопку «Старт». Данная кнопка запускает механизм сбора данных для списка адресов пользователей, собранных механизмом сбора адресов. Процесс работы продемонстрирован на рисунке 17.

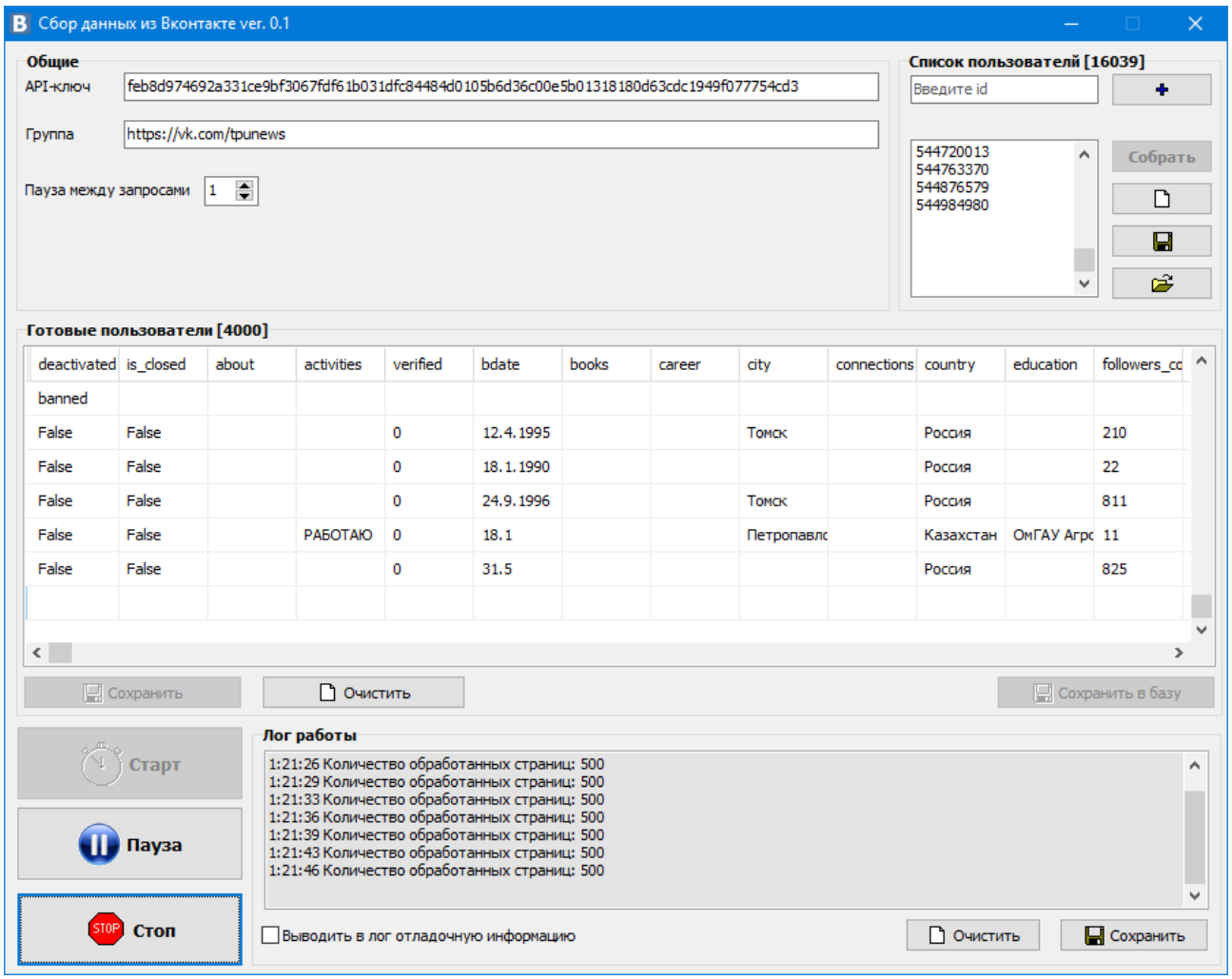


Рисунок 17 - Процесс работы механизма сбора данных

Сбор данных по списку адресов страниц всех 20539 участников исследуемой группы занял 118 секунд при выставлении соответствующего параметра «Пауза между запросами» на минимум (1 секунда). Рисунок 18 демонстрирует успешное завершение механизма сбора данных.

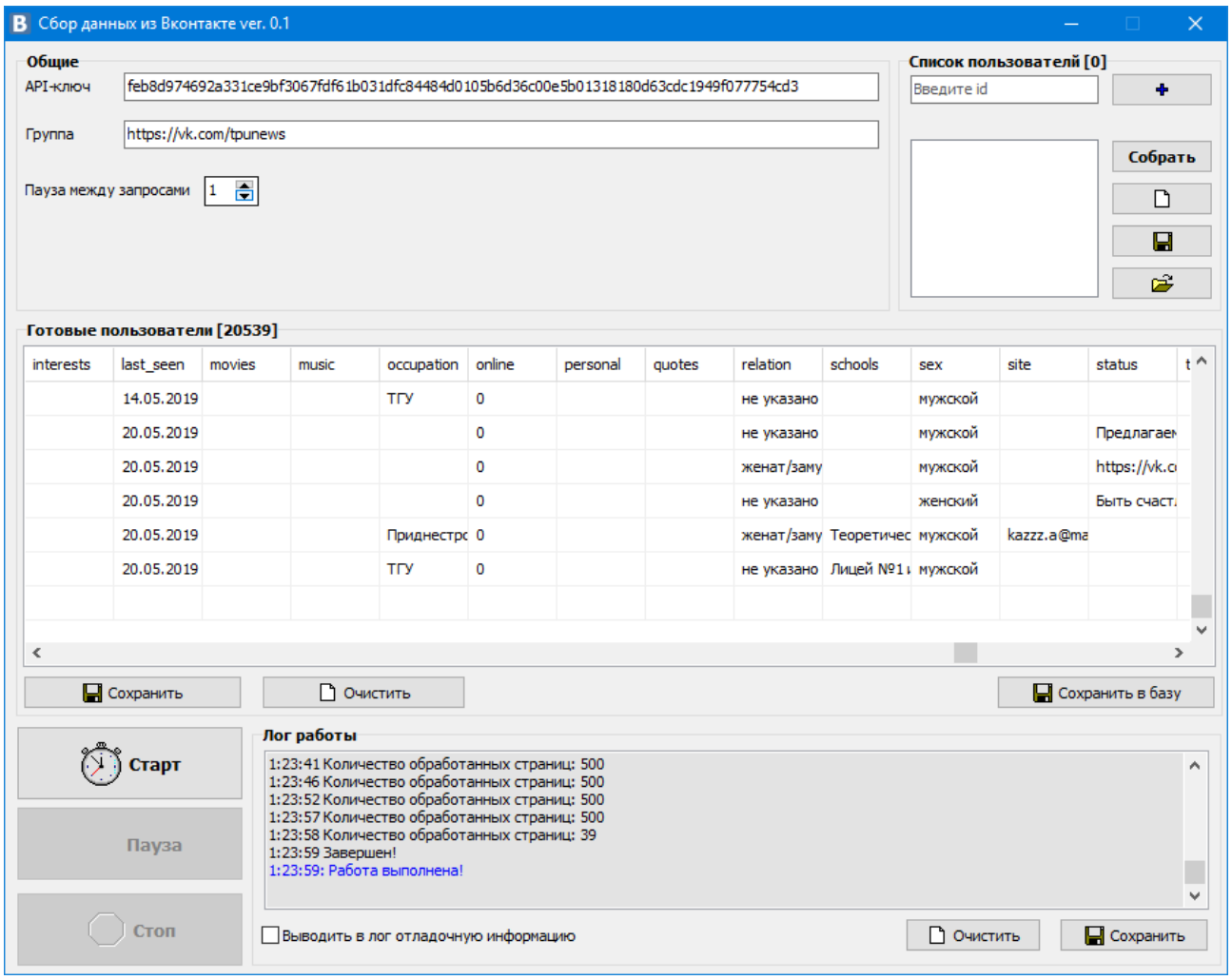


Рисунок 18 - Завершение работы механизма сбора данных

Для конвертации таблицы с собранными программой данными в формат Excel .xls была нажата кнопка «Сохранить», расположенная в одном интерфейсном блоке с таблицей. Результат успешного сохранения данных в файл Excel представлен на рисунке 19.

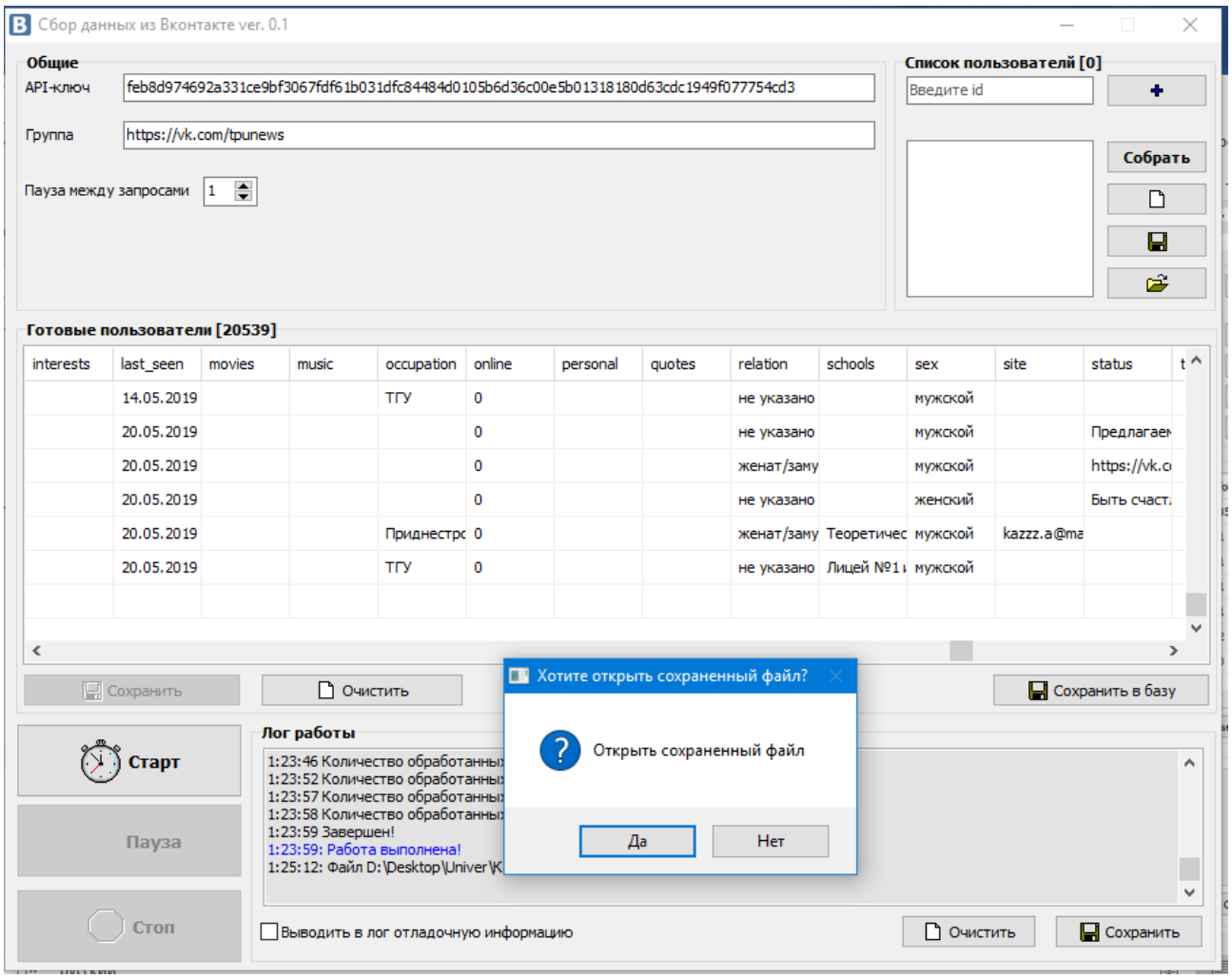
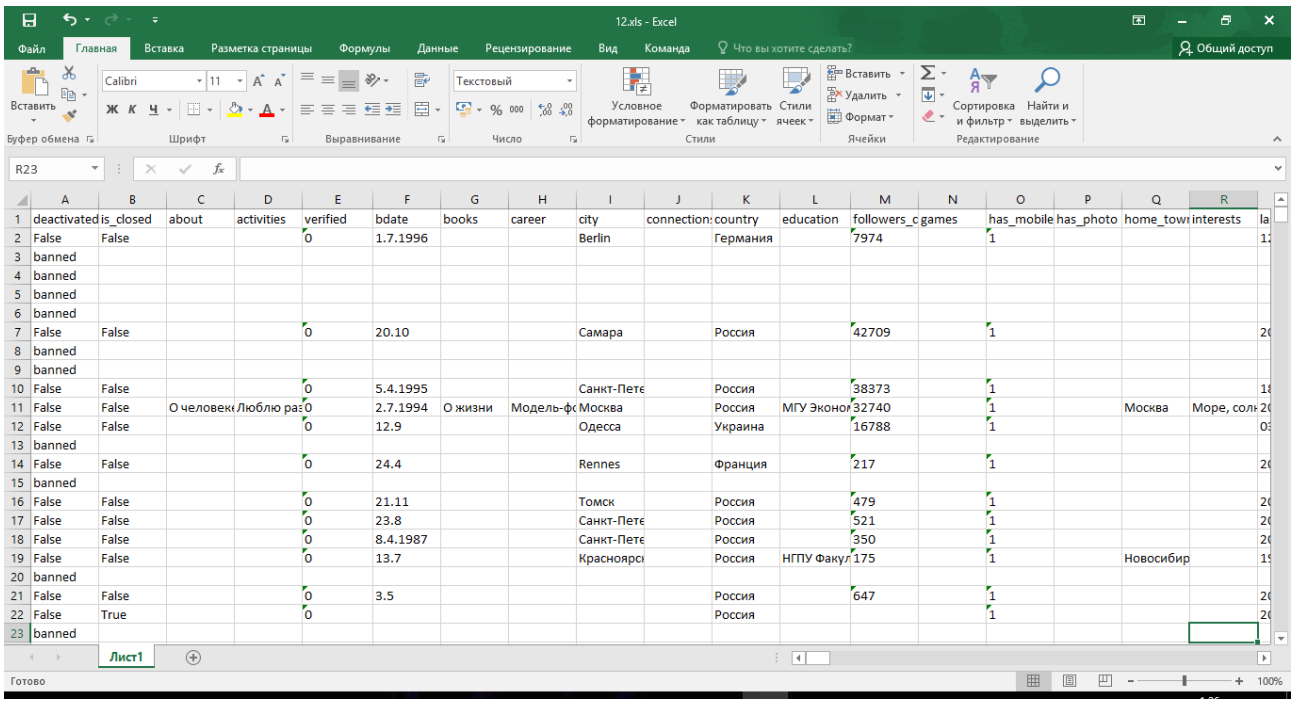


Рисунок 19 - Результат сохранения данных в формат Excel

В созданном Excel-файле размещаются все собранные данные. Содержимое созданного файла представлено на рисунке 20. Для каждого столбца сгенерирован соответствующий заголовок.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	deactivated	is_closed	about	activities	verified	bdate	books	career	city	connection	country	education	followers_c	games	has_mobile	has_photo	home_tow	interests
2	False	False			0	1.7.1996			Berlin		Германия		7974		1			
3	banned																	
4	banned																	
5	banned																	
6	banned																	
7	False	False			0	20.10			Самара		Россия		42709		1			
8	banned																	
9	banned																	
10	False	False			0	5.4.1995			Санкт-Пете		Россия		38373		1			
11	False	False	О человеке	Люблю раз	0	2.7.1994	О жизни	Модель-фо	Москва		Россия	МГУ Эконом	32740		1		Москва	Море, соль
12	False	False			0	12.9			Одесса		Украина		16788		1			
13	banned																	
14	False	False			0	24.4			Rennes		Франция		217		1			
15	banned																	
16	False	False			0	21.11			Томск		Россия		479		1			
17	False	False			0	23.8			Санкт-Пете		Россия		521		1			
18	False	False			0	8.4.1987			Санкт-Пете		Россия		350		1			
19	False	False			0	13.7			Красноярск		Россия	НГПУ Факул	175		1		Новосибир	
20	banned																	
21	False	False			0	3.5					Россия		647		1			
22	False	True			0						Россия				1			
23	banned																	

Рисунок 20 - Содержимое сохраненного Excel файла с данными

Для сохранения собранных данных в базу данных с форматом .mdb была нажата кнопка «Сохранить в базу». Результат поведения программы после нажатия представлен на рисунке 21.

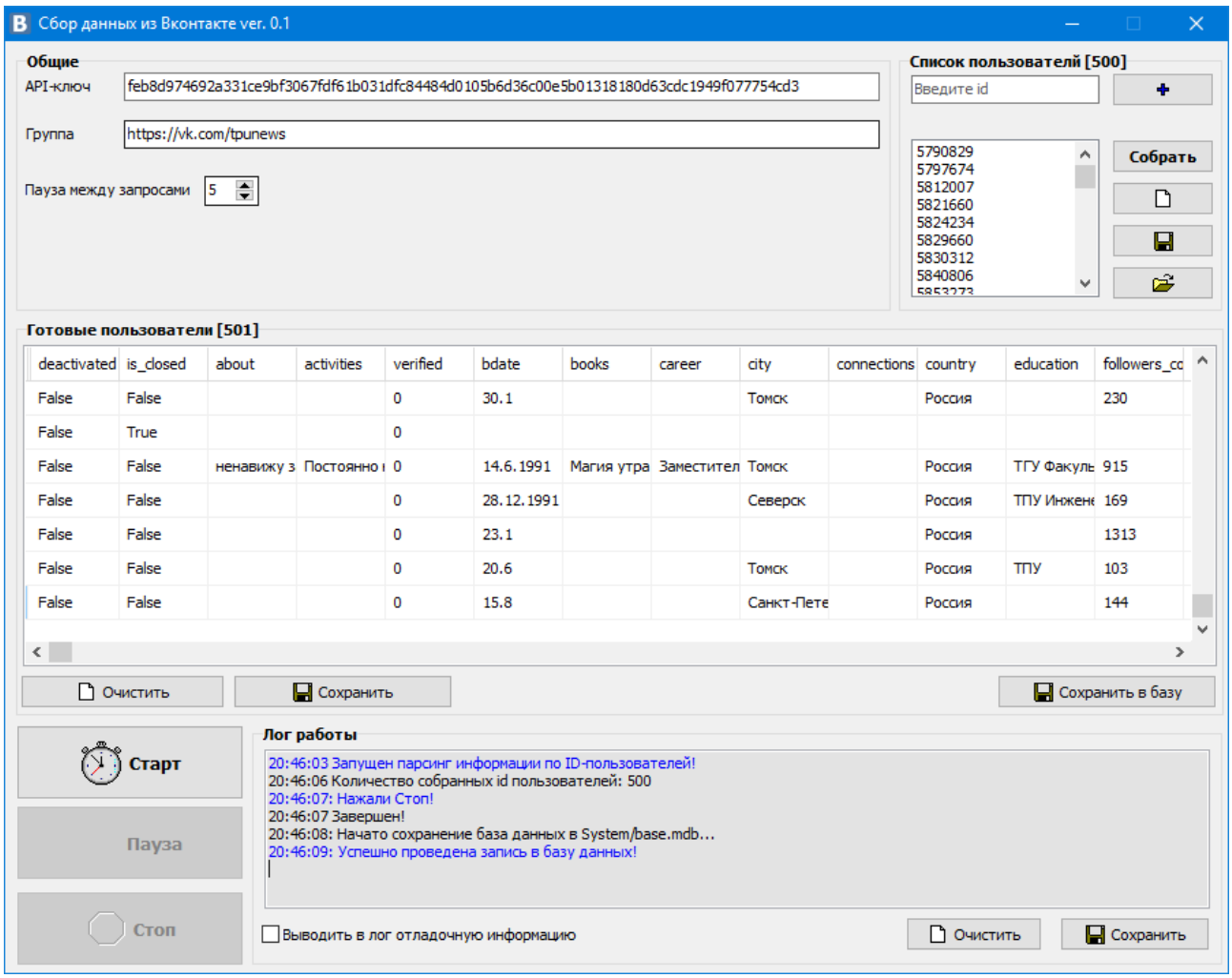


Рисунок 21 - Процесс сохранения данных в БД

База данных по содержанию и строению идентична созданному документу Excel.

Глава 4. Финансовый менеджмент, ресурсоэффективность и ресурсосбережение

В данной работе осуществляется разработка программной системы для сбора неперсональных данных из социальной сети «Вконтакте». Задача раздела: экономически обосновать данной разработку, определить и рассчитать трудовые и денежные затраты на её создание.

4.1. Оценка коммерческого потенциала и перспективности проведения научных исследований с позиции ресурсоэффективности и ресурсосбережения

4.1.1. Потенциальные потребители результатов исследования

Разрабатываемая программная система предназначена для продажи пользователям, которые в этом заинтересованы, например, социологам, маркетологам, администраторам групп, владельцам малого, среднего и большого бизнеса, а также прочим заинтересованным в этом личностям и организациям. Данная программная система предназначена для улучшения качества и значительного уменьшения временных затрат сбора неперсональной информации о пользователях социальной сети «Вконтакте».

4.1.2. Анализ конкурентных технических решений

В ходе выполнения работы были выявлены и проанализированы конкуренты разрабатываемой программной системы. По результатам анализа была составлена таблица № 1 «Оценочная карта для сравнения конкурентных технических решений (разработок)» [17].

Таблица 1 – Оценочная карта для сравнения конкурентных технических решений (разработок)

№ п/п	Конкуренты	Факторы конкурентоспособности (по 10-бальной шкале)					Итоговая оценка
		Доступность ПО по стоимости	Скорость сбора данных	Поддерживаемые социальные сети	Форматы сохранения	Максимум запросов в день	
1	Agora Pulse	3 (от 17449 до 33145 руб. в месяц)	6 (8 секунд за 1000 пользователей)	7 («Facebook», «Twitter», «Instagram»,	4 (БД)	5 (1000 ед.)	15 + 24 + 28 + 12 + 25 = 104

				«LinkedIn», «YouTube»)			
2	YouScan	1 (от 77458 до 155022 руб. в месяц)	7 (6 секунд за 1000 пользователей)	7 («Facebook», «Twitter», «YouTube», «Вконтакте»)	5 (XLSX, PDF, PPTX)	10 (неограниченно)	5 + 28 + 28 + 26 + 50 = 137
3	Разработанный проект	8 (5000 руб. в месяц)	7 (6 секунд за 1000 пользователей)	4 («Вконтакте»)	6 (БД, XLSX)	10 (неограниченно)	40 + 28 + 16 + 18 + 50 = 152
	b_j	5	4	4	3	5	
	w_j	12	20	18	15	25	-

Анализируя данные из вышеприведенной таблицы можно сделать вывод о том, что разрабатываемая программная превосходит все конкурентные решения по следующим факторам: доступность ПО по стоимости, скорость сбора данных, форматы сохранения, максимум запросов в день.

На рисунке 22 представлен многоугольник конкурентоспособности [18] с учетом важности факторов.

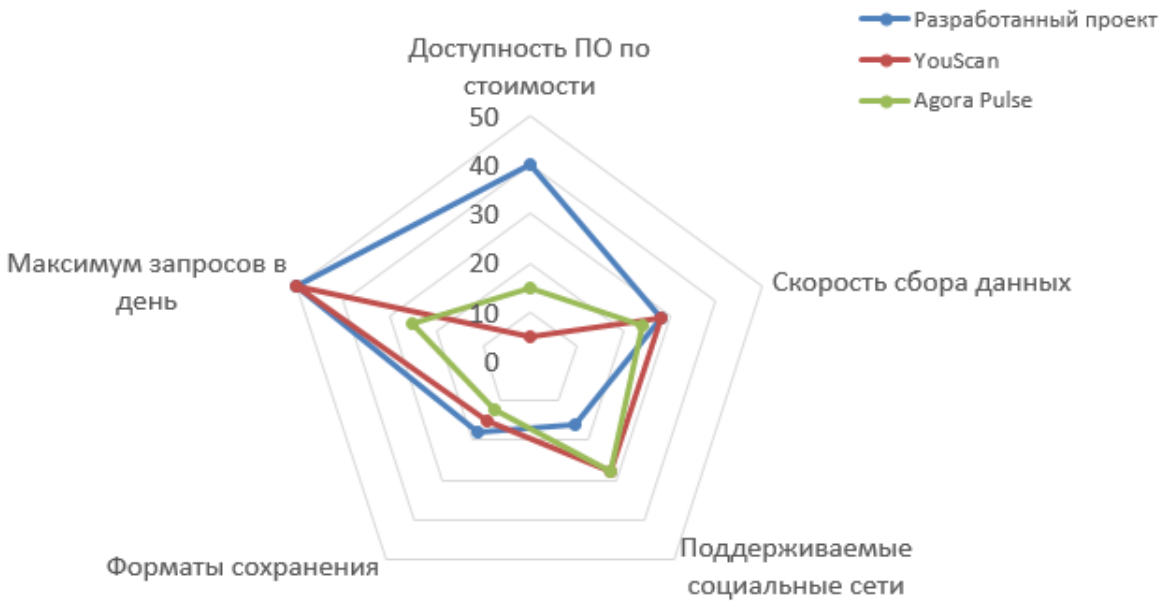


Рисунок 22 - Многоугольник конкурентоспособности

Самой сильной стороной разрабатываемого ПО является доступная стоимость, а самой слабой – количество поддерживаемых социальных сетей.

4.1.3. SWOT-анализ

SWOT-анализ – один из самых часто используемых методов анализа в менеджменте и маркетинге [19]. Данным метод дает ясное представление о текущей ситуации, а также помогает понять, какие действия необходимо предпринять для максимизации возможностей проекта и нейтрализации слабых сторон и угроз.

Целью использования SWOT-анализа для данной разработки является определение возможной эффективности и прогнозирование направлений будущего развития разрабатываемого решения.

Преимуществом SWOT-анализа является разработка связей разнообразных факторов внешней и внутренней среды разработки.

Результаты проведения SWOT-анализа представлены в сводной таблице 2, где указаны сильные и слабые стороны разработки, выявлены возможные направления будущей разработки программной системы и рассмотрены варианты минимизации влияния угроз.

Таблица 2. Сводная матрица SWOT-анализа

	Сильные стороны проекта: С1. Доступная стоимость ПО. С2. Высокая скорость работы ПО. С3. Вариативность при сохранении данных. С4. Неограниченное количество запросов в день.	**Слабые стороны проекта:** Сл1. Зависимость от социальной сети «Вконтакте». Сл2. Необходимо постоянное наличие подключения к сети Internet. Сл3. Отсутствие проверок на наличие лицензии ПО.
Возможности: В1. Сбор неперсональных данных из других социальных сетей. В2. Увеличение скорости сбора данных.	1) Разработка функционала по сбору данных из других социальных сетей. 3) Разработка новых продуктов для поддержания позиций на рынке ПО.	1) Разработка новых способов сбора данных и осуществление сбора из других социальных сетей. 2) Реализовать систему контроля лицензирования ПО. 3) Реализовать защиту ПО от взлома.
Угрозы: У1. Социальная сеть «Вконтакте» закроет API для запросов внешних систем. У2. Противодействие со стороны конкурентов.	1) Продвижение ПО с акцентированием на достоинствах. 2) Доработка ПО. 3) Снижение цен на ПО.	1) Анализ ситуации на рынке с выявлением наилучших стратегий и решений. 2) Проработка функционала и новых способов сбора данных. 3) Проведение маркетинговых кампаний, снижение стоимости ПО и добавление акций на покупку ПО.

4.2. Планирование научно-исследовательских работ

Рациональное планирование занятости участников разработки и сроков проведения каждого из этапов работы позволяет успешно организовать процесс работы над конкретной задачей.

4.2.1. Структура работ в рамках научного исследования

В данном этапе был составлен список работ, а также назначен исполнитель и выставлена временная продолжительность. Результатом планирования работ является линейных график реализации проекта.

Перечень этапов работы и распределение исполнителей представлен в таблице 3.

Таблица 3. Перечень этапов работы и распределение исполнителей

№ п/п	Этапы работы	Исполнители
1	Постановка целей и задач	Научный руководитель
2	Разработка и утверждение ТЗ	Научный руководитель, Студент
3	Подбор и изучение материалов по тематике	Научный руководитель, Студент
4	Разработка календарного плана	Научный руководитель, Студент
5	Обсуждение литературы	Научный руководитель, Студент
6	Проведение анализа предметной области	Студент
7	Проектирование	Научный руководитель, Студент
8	Разработка	Студент
9	Тестирование и отладка	Студент
10	Согласование выполненной работы с научным руководителем	Студент
11	Оформление работы	Студент

4.2.2. Определение трудоемкости выполнения работ и разработка графика проведения научного исследования

Трудовые затраты являются основными затратами на разработку, поэтому необходимо определить их для каждого из исполнителей.

Ожидаемая продолжительность работ $t_{ож}$ с помощью экспертных оценок устанавливается согласно формуле:

$$t_{ожi} = \frac{3t_{\min_i} + 2t_{\max_i}}{5},$$

где t_{min} – минимальная продолжительность работ в днях;

t_{max} – максимальная продолжительность работ в днях.

$$t_{ож1} = \frac{3 \cdot 2 + 2 \cdot 4}{5} = 2.8 \; дн.$$

Остальные значения были рассчитаны аналогичным способом.

Формула для вычисления длительности этапов в рабочих днях $Т_{РД}$:

$$Т_{РД} = t_{ож} \cdot К_{Д},$$

где $К_Д$ – коэффициент, который учитывает дополнительное время на компенсации и согласование работ. Он равен 1,2.

$Т_{РД1}$ равен 3,36 дн., остальные значения рассчитаны по аналогии.

Продолжительность этапа в рабочих днях $Т_{КД}$ рассчитывалась по формуле:

$$Т_{КД} = Т_{РД} \cdot Т_К ,$$

где $Т_{РД}$ – продолжительность выполнения этапа в рабочих днях;

$Т_К$ – коэффициент календарности, вычисляется по формуле:

$$Т_к = \frac{Т_{кал}}{Т_{кал} - Т_{вд} - Т_{пд}}$$

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

$Т_{ВД}$ – выходные дни ($Т_{ВД}$ = 53);

$Т_{ПД}$ – праздничные дни ($Т_{ПД}$ = 14).

Коэффициент календарности $Т_К$ равен 1,244. Продолжительность первого этапа в календарных днях $Т_{КД1}$ равна 4,18. Остальные значения рассчитаны аналогично.

Все расчеты по трудозатратам представлены в таблице 4, результаты продолжительности этапов работы являются общими трудоемкостями для каждого из исполнителей проекта.

По величинам трудоемкости этапов по исполнителям $T_{КД}$ (данные столбцов 9 и 10) был построен линейный график реализации проекта. Данный график приведен в таблице 4.

Таблица 3. Трудозатраты на выполнение проекта

Этап работы	**Исполнители, %**		**Продолжительность работ, дни**			**Длительность работ, чел/дн.**			
						$T_{РД}$		$T_{КД}$	
	Науч. рук.	**Студ.**	t_{min}	t_{max}	$t_{ож}$	**НР**	**И**	**НР**	**И**
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Постановка целей и задач	100	0	2	4	2,8	3,36	0	4,18	0,00
Разработка и утверждение ТЗ	50	50	4	6	4,8	2,88	2,88	3,58	3,58
Подбор и изучение материалов по тематике	70	30	5	10	7	5,88	2,52	7,31	3,13
Разработка календарного плана	50	50	2	4	2,8	1,68	1,68	2,09	2,09
Обсуждение литературы	50	50	2	4	2,8	1,68	1,68	2,09	2,09
Проведение анализа предметной области	0	100	4	6	4,8	0	5,76	0,00	7,17
Проектирование	30	70	6	10	7,6	2,736	6,384	3,40	7,94
Разработка	0	100	30	40	34	0	40,8	0,00	50,76
Тестирование и отладка	0	100	8	12	9,6	0	11,52	0,00	14,33
Согласование выполненной работы с научным руководителем	0	100	6	10	7,6	0	9,12	0,00	11,35
Оформление работы	10	90	4	6	4,8	0,576	5,184	0,72	6,45
Итого:					**88,6**	**18,79**	**87,53**	**23,38**	**108,88**

Таблица 4. Линейный график работ

Этап	Ткд НР, кал. дн.	Ткд Ст, кал. дн.	Продолжительность выполнения работ													
			январь			февраль			март			апрель			май	
			10	20	30	40	50	60	70	80	90	100	110	120	130	140
1	4,18	0,00														
2	3,58	3,58														
3	7,31	3,13														
4	2,09	2,09														
5	2,09	2,09														
6	0,00	7,17														
7	3,40	7,94														
8	0,00	50,76														
9	0,00	14,33														
10	0,00	11,35														
11	0,72	6,45														

Примечание: –Руководитель – Студент

4.2.3. Бюджет проекта

Для проекта по разработке программной системы по сбору неперсональной информации из социальной сети «Вконтакте» производится оценка затрат по следующим статьям:

- материалы;
- заработная плата;
- отчисления во внебюджетные фонды;
- расходы на электроэнергию (без освещения);
- амортизационные расходы;
- накладные расходы.

Работа по проекту выполнялась без участия иных организаций и без командировок и аренды имущества, следовательно, расходы по соответствующим статьям отсутствуют.

4.2.3.1. Расчет затрат на материалы

В затратах на материалы были учтены затраты на бумагу и картриджи для принтера, поскольку все необходимые материалы имелись в инвентаре исполнителей. Расчет затрат на материалы представлен в таблице 5.

Таблица 5. Затраты на материалы

Наименование материалов	**Цена за ед., руб.**	**Кол-во**	**Сумма, руб.**
Бумага для принтера, А4	240,00	1 уп.	240,00
Картридж	1400,00	1 шт.	1400,00
Итого:			**1640,00**

4.2.3.2. Расчет заработной платы

Расчет основной заработной платы выполняется на основе трудоемкости выполнения каждого этапа и величины месячного оклада исполнителя.

Месячный оклад (МО) научного руководителя, занимающего должность доцента и имеющего степень кандидата технических наук, составляет 33664 руб./мес., МО исполнителя составляет 21760 руб./мес.

Исходя из того, что в месяце в среднем 24,83 рабочих дня при шестидневной рабочей неделе, среднедневная тарифная заработная плата ($ЗП_{дн-т}$) рассчитывается по формуле:

$$ЗП_{дн-т} = МО/24,83$$

Расчеты затрат на полную заработную плату приведены в таблице 6. Затраты времени по каждому исполнителю, в рабочих днях с округлением до целого, взяты из таблицы 3, где указаны трудозатраты исполнителей. Для учета в ее составе премий, дополнительной зарплаты и районной надбавки используется районный коэффициент $К_р = 1,3$.

Таким образом, для перехода от тарифной суммы заработка исполнителя, связанной с участием в проекте, к соответствующему полному заработку необходимо учесть районный коэффициент $К_р = 1,3$.

Таблица 6. Затраты на заработную плату

Исполнитель	**Оклад, руб./мес.**	**Среднедневная ставка, руб./раб.день**	**Затраты времени, раб.дни**	**$К_Р$**	**Фонд з/платы, руб.**
Научный руководитель	33664,00	1355,78	24	1,3	42300,34
Студент	21760,00	876,36	109	1,3	124180,21
Итого:					**166480,55**

4.2.3.3. Расчет затрат на отчисления во внебюджетные фонды

Отчисления во внебюджетные фонды включают в себя отчисления в пенсионный фонд, на социальное и медицинское страхование и составляют 28% от заработной платы участников проекта.

$С_{соц}$ определяется следующим образом:

$$C_{соц.} = C_{зп} * 0,28 = (42300,34 + 124180,21) * 0,28 = 46614,55 \text{ руб.}$$

4.2.3.4. Расчет затрат на электроэнергию

Данный вид расходов включает в себя затраты на электроэнергию при работе оборудования, а именно компьютера и принтера. Затраты на электроэнергию при работе оборудования $C_{эл.об.}$ рассчитываются по формуле:

$$С_{эл.об.} = P_{об} \cdot Ц_{э} \cdot t_{об},$$

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

$Ц_э$ – тарифная цена за 1 кВт·час; $t_{об}$ – время работы оборудования, час.

Мощность $P_{об}$, потребляемая оборудованием, определяется по формуле:

$$P_{об} = P_{ном.} \cdot K_{С},$$

где $P_{ном.}$ – номинальная мощность оборудования, кВт;

$K_С$ – коэффициент загрузки (для технологического оборудования малой мощности $K_С = 1$).

Номинальная мощность персонального компьютера составляет 0,3 кВт, принтера – 0,1 кВт.

С учетом налога на добавленную стоимость (НДС) $Ц_э = 5,257$ руб./кВт·час.

Время работы оборудования $t_{ОБ}$ для исполнителя вычисляется на основе данных таблицы 3, где указаны трудозатраты проекта:

$$t_{ОБ} = T_{РД} \cdot K_t,$$

где $К_t \leq 1$– коэффициент использования оборудования по времени, равный отношению времени его работы в процессе выполнения проекта к $T_{РД}$.

Из расчета, что продолжительность рабочего дня равна 8 часов, а работа выполнялась 109 рабочих дней, получим, что общее время выполнения проекта составляет 872 часа, которые были проведены за компьютером

Принтер использовался в течение 1 часа. Затраты на электроэнергию при работе оборудования сведены в таблицу 7.

Таблица 7. Затраты на электроэнергию для технологических целей

Наименование оборудования	**Время работы оборудования $t_{ОБ}$, час**	**Потребляемая мощность $Р_{ОБ}$, кВт**	**Затраты $Э_{ОБ}$, руб.**
Персональный компьютер	872	0,3	1375,23
Лазерный принтер	1	0,1	0,53
Итого			1375,76

4.2.3.5. Расчет амортизационных расходов

Амортизационные отчисления для рассматриваемого проекта включают в себя амортизацию используемого оборудования за время выполнения работы. Они рассчитываются по времени использования компьютера, по следующей формуле:

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

где: $Н_А$ – годовая норма амортизации; $Ц_{ОБ}$ – цена оборудования;

$F_Д$ – действительный годовой фонд рабочего времени;

$t_{рф}$ – время работы вычислительной техники;

n – число задействованных единиц оборудования, n = 1.

Годовая амортизация $Н_А$ – величина, обратная сроку амортизации оборудования $С_А$, который определяется согласно постановлению правительства РФ «О классификации основных средств, включенных в амортизационные группы». Для компьютера примем $С_А$ = 3 года, тогда $Н_А$ = 0,33. Для принтера примем $С_А$ = 2 года, тогда $Н_А$ = 0,5.

Расчет затрат на амортизационные отчисления представлен в таблице 8.

Таблица 8. Затраты на амортизационные отчисления

Наименование оборудования	**Норма аморт. оборуд., $Н_А$**	**Стоим. оборуд., $Ц_{ОБ}$, руб.**	**Факт. р/вр. оборуд., $t_{рф}$, ч**	**Действ. год. фонд р/вр., $F_Д$, ч.**	**Аморт. отчисл., $С_{АМ}$, руб.**
Персональный компьютер	0,33	30000,00	872	2384	3621,14
Лазерный принтер	0,50	12000,00	1	2384	2,52
Итого					3623,66

4.2.3.6. Расчет накладных расходов

В данном разделе были рассчитаны накладные расходы. Величина накладных расходов составляет 16% от суммы всех указанных ранее затрат и рассчитывается по формуле:

$$С_{проч} = 0{,}16 \cdot (С_{мат} + С_{зп} + С_{соц} + С_{эл} + С_{ам})$$

$$С_{проч} = 0{,}16 \cdot (1640{,}00 + 166480{,}55 + 46614{,}55 + 1375{,}76 + 3623{,}66) = 35157{,}52$$

Таким образом, прочие накладные расходы составили 35157,52 руб.

4.2.3.6. Расчет общей себестоимости разработки

Общая себестоимость разработки программной системы по сбору неперсональных данных из социальной сети «Вконтакте» рассчитывается с помощью суммирования всех расходов. Она представлена в таблице 9.

Таблица 9. Общая себестоимость разработки проекта

Статья затрат	**Условное обозначение**	**Сумма, руб.**
Материалы и покупные изделия	$С_{мат}$	1640,00
Заработная плата	$С_{зп}$	166480,55
Отчисления в социальные фонды	$С_{соц}$	46614,55
Расходы на электроэнергию	$С_{эл.об.}$	1375,76

Амортизационные отчисления	$Э_{ам}$	4949,20
Накладные расходы	$C_{накл}$	35157,52
Итого:		**256217,58**

Общая себестоимость проекта получилась равной 256217,58 рублей. Наиболее сильно на общую стоимость влияют такие статьи затрат, как «Заработная плата», «Отчисления в социальные фонды» и «Накладные расходы».

4.3. Определение ресурсной (ресурсосберегающей), финансовой, бюджетной, социальной и экономической эффективности исследования

Потенциальными пользователями разработанной программной системы по сбору неперсональных данных из социальной сети «Вконтакте» являются социологи, маркетологи, SMM-специалисты, администраторы групп, владельцы малого, среднего и большого бизнеса, а также прочие заинтересованные личности и организации.

Общая длительность разработки программной системы составляет 109 дней. Общая себестоимость проекта описана в разделе «Расчет общей себестоимости разработки» и составляет 256217,58 рублей.

Стоимость продажи программной системы пользователям описана в разделе «Описание разрабатываемой программной системы» и составляет 5000 рублей в месяц (60000 рублей в год) за одну единицу программного обеспечения.

В разделе «Анализ конкурентных технических решений» были рассмотрены конкуренты на рынке программного обеспечения, а именно «Agora Pulse» и «YouScan», которые в общей итоговой оценке набрали 107 и 134 баллов соответственно, а разработанная программная система набрала 152 баллов, что превосходит количество баллов у «Agora Pulse» на 42%, а количество баллов у «YouScan» - на 13%.

Количество покупателей аналогичного решения у «Agora Pulse» составляет около 11000 человек, а у «YouScan» - 2000 человек.

Для того, чтобы окупить разработку, программу должны приобрести на год не менее 5 человек, что является преодолимой целью ввиду того факта, что разработанная программная система значительно превосходит вышеупомянутых конкурентов по общей итоговой оценке.

4.4. Выводы

В ходе выполнения раздела финансового менеджмента проведен анализ финансово-экономических аспектов разработки программной системы. Составлен перечень проводимых работ, их исполнителей и продолжительность выполнения этапов работ, составлен линейный график.

Также произведен расчет сметы затрат на выполнение проекта, проведен расчет себестоимости проекта, определены показатели эффективности проекта и проведена оценка его эффективности.

Глава 5. Социальная ответственность

5.1. Введение

Трудовая деятельность разработчика программных систем связана с воздействием производственных факторов различного характера. Для предупреждения вредного воздействия и сохранения здоровья работника предусмотрены различные меры по обеспечению безопасности трудовой деятельности.

В разделе проведен анализ вредных и опасных факторов труда, определен комплекс мер организационного, правового, технического и режимного характера, который должен способствовать снижению возможности возникновения негативных последствий работы разработчика.

Выпускная квалификационная работа по разработке программной системы по сбору неперсональных данных из социальной сети «Вконтакте» выполнялась в ходе преддипломной практики в Кибернетическом центре. Рабочее место является офисным помещением, где будет работать разработчик.

Характеристика помещения:

- ширина помещения – 6 м, длина – 7 м, высота – 3 м;
- площадь помещения – 42 $м^2$;
- объем помещения – 126 $м^3$;
- в помещении установлен кондиционер, имеется естественная вентиляция – вытяжное вентиляционное отверстие, дверь, окно;
- в помещении установлено искусственное освещение, имеется естественное освещение.

В данном помещении оборудовано три рабочих места, максимальное количество сотрудников в одну смену – 3. В среднем, на одного сотрудника приходится 14 $м^2$ площади и около 42 $м^3$ объема помещения. Данное размещение сотрудников удовлетворяет санитарным нормам, согласно которым на одного работника должно приходиться не менее 6 $м^2$ площади и 24 $м^3$

объема рабочего помещения, с учетом максимального количества сотрудников, одновременно работающих в смену.

5.2 Правовые и организационные вопросы обеспечения безопасности

При организации рабочего места с персональными компьютерами необходимо учитывать требования безопасности, промышленных санитарных норм, эргономики и технической эстетики.

Рабочее место должно быть организовано с учетом требований ГОСТ 12.2.032-78 «ССБТ. Рабочее место при выполнении работ сидя. Общие эргономические требования» [24] и СанПиН 2.2.2/2.4.1340-03 «Гигиенические требования к персональным электронно-вычислительным машинам и организации работы» [25].

Согласно требованиям, при организации работы с ПК должны выполняться следующие условия:

- площадь на одно рабочее место пользователя с ПК должна составлять не менее 6 м²;
- конструкция рабочей мебели должна регулироваться соответственно росту пользователя;
- Персональный компьютер и рабочее место должны располагаться так, чтобы свет падал сбоку;
- расстояние от персонального компьютера до стен должно быть более 1 м, по возможности следует избегать расположения рабочих мест в углах помещения, либо лицом к стене;
- Персональный компьютер следует установить так, чтобы, подняв глаза от экрана, можно было увидеть удаленный предмет в помещении или на улице, таким образом, предоставляя эффективный способ разгрузки зрительного аппарата;

- окна в помещениях с ПК должны быть оборудованы регулирующими устройствами – жалюзи, занавески, внешние козырьки;
- монитор, клавиатура и корпус компьютера должны находиться прямо перед работником;
- высота рабочего стола с клавиатурой должна составлять 680-800 мм над уровнем стола;
- высота экрана над полом – 900-1280 мм, монитор должен находиться на расстоянии 600-700 мм от работника на 20 градусов ниже уровня глаз;
- рабочее кресло должно иметь мягкое сиденье и спинку, с регулировкой сиденья, с удобной опорой для поясницы
- положение тела пользователя относительно монитора должно соответствовать направлению просмотра под прямым углом 90 градусов или под углом 75 градусов.

В соответствии с Трудовым кодексом РФ 197-ФЗ предусмотрена рациональная организация труда в течение смены, согласно которой:

- длительность рабочей смены должна быть не более 8 часов;
- должны быть установлены два регламентируемых перерыва - не менее 20 минут после 1-2 часов работы или не менее 30 минут после 2 часов работы;
- обеденный перерыв должен быть не менее 40 минут, может быть скользящим в течение рабочей смены.

Также, Трудовым кодексом закреплен обязательный предварительный медицинский осмотр при приеме на работу и периодические медицинские осмотры.

Прохождение инструктажа по технике безопасности должен проходиться каждым работником перед трудоустройством, а после должен быть пройден инструктаж по электробезопасности и охране труда. Каждому

работнику обязательно должна быть предоставлена рабочая инструкция, с описанием входящих в его должность функций и рабочих моментов, а также конкретным описанием границ ответственности.

При выполнении ВКР на представленном рабочем месте нарушения правовых и организационных норм не было, рабочее место оборудовано согласно санитарным и эргономическим нормам, организация рабочего времени согласно регламентированным нормам.

5.3. Производственная безопасность

При анализе работы химические и биологические факторы не оказывают существенного влияния на состояние здоровья разработчика программных систем, поэтому рассмотрим только физические и психофизиологические факторы.

Перечень опасных и вредных факторов представлен в таблице 10.

Таблица 10. Перечень опасных и вредных факторов (ГОСТ 12.0.003-2015)

Факторы (ГОСТ 12.0.003-2015)	Этапы работ			Нормативные документы
	Проектирование	Разработка	Тестирование	
Повышенные уровни электромагнитных излучений	+	+	+	СанПиН 2.2.4/2.1.8.055-96. «Электромагнитные излучения радиочастотного диапазона» [26]
Отклонение показателей микроклимата	+	+	+	СанПиН 2.2.4.548-96 «Гигиенические требования к микроклимату производственных помещений» [27]
Недостаточная	+	+	+	СанПиН 2.2.1/2.1.1.1278-03

освещенность рабочей зоны				«Гигиенические требования к естественному, искусственному и совмещенному освещению жилых и общественных зданий» [28]
Превышение уровня шума	+	+	+	СН 2.2.4/2.1.8.562-96 «Шум на рабочих местах, в помещениях жилых, общественных зданий и на территории жилой застройки» [29]
Повышенное значение напряжения в электрической цепи, замыкание которой может произойти через тело человека	+	+	+	ГОСТ Р 12.1.019-2009 «Система стандартов безопасности труда (ССБТ). Электробезопасность. Общие требования и номенклатура видов защиты» [30]

Проведем анализ всех вышеперечисленных факторов и определим соответствие рабочего места установленным санитарным нормам.

5.3.1. Повышенные уровни электромагнитных излучений

Главным рабочим аппаратом разработчика программных систем является персональный компьютер (ПК), который в период работы подвергает работника вредному электромагнитному излучению. Электромагнитное излучение ПК сложное по спектральному составу, изменяется в диапазоне частот от 0 Гц до 1000 МГц. Такое излучение состоит из электрической (Е) и магнитной (Н) составляющих.

Норма допустимых уровней напряженности полей и излучений регламентируются СанПиН 2.2.4.1191-03 и нормами Госкомсанэпидемнадзора «Гигиенические требования к персональным электронно-вычислительным машинам и организации работы» (СанПиН 2.2.4.1340-03) [21, 22]. По установленным нормам время пребывания работника в рабочей зоне вычисляется по формуле:

$$T = (50/E) – 2.$$

Так, например, при напряженности до 5 кВ/м присутствие работника в рабочей зоне разрешается в течение 8 часов, а при напряженности 20-25 кВ/м время присутствия работника в рабочей зоне сокращается до 10 минут.

Воздействие электромагнитного излучения на человека негативно влияет на такие органы, как мозг, глаза, желудок, почки. Облучение глаз может привести к катаракте или помутнению хрусталика глаза. При длительном воздействии электромагнитного излучения может произойти нервное расстройство.

Методы и средства для минимизации воздействия фактора: применение специальных приборов-минимизаторов электромагнитного излучения и сокращение времени пребывания рабочего в области воздействия данного фактора.

В предполагаемом рабочем месте уровень напряженности электрических полей не превышает значения 4 кВ/м, при котором разрешенное время пребывания в рабочей зоне может составлять до 10,5 часов. Рабочая смена длится 8 часов, следовательно, уровень электромагнитных излучений на рабочем месте в норме.

5.3.2. Отклонение показателей микроклимата

Под микроклиматом рабочего помещения понимают климат внутренней среды помещения, в котором находятся сотрудники в течение рабочего

времени. Микроклимат определяется совокупностью показателей, воздействующих на организм работника. Они регламентируются СанПиН 2.2.4.548-96 «Гигиенические требования к микроклимату производственных помещений» [23]. Санитарные нормы устанавливают оптимальные и допустимые значения показателей в рабочей зоне, что позволяет создавать благоприятные условия работы, соответствующие физиологическим потребностям организма человека. Для поддержания и доведения микроклиматических показателей до нормативных значений проводятся мероприятия, которые обязательно должны включаться в комплексные планы предприятий по охране труда.

При отклонении показателей микроклимата у работника могут возникнуть такие последствия, как переохлаждение, перегревание, заболевания легких.

Работа, выполняемая разработчиком программных систем, относится к категории Iа, так как она является малоподвижной и малоинтенсивной, проводится сидя с минимальными физическими напряжениями. В таблицах 11 и 12 представлены оптимальные и допустимые значения показателей микроклимата на рабочих местах для данной категории.

Таблица 11. Оптимальные величины показателей микроклимата на рабочих местах производственных помещений (СанПиН 2.2.4.548-96)

Период года	**Температура воздуха, °С**	**Температура поверхностей, °С**	**Относительная влажность воздуха, %**	**Скорость движения воздуха, м/с**
Холодный	22-24	21-25	60-40	0,1
Теплый	23-25	22-26	60-40	0,1

Таблица 12. Допустимые величины показателей микроклимата на рабочих местах производственных помещений (СанПиН 2.2.4.548-96)

Период	**Температура**	**Темпер**	**Относи**	**Скорость движения воздуха,**

года	воздуха, °С		атура поверхностей, °С	тельная влажность воздуха, %	м/с	
	диапазон ниже оптимальных величин	диапазон выше оптимальных величин			для диапазона температур воздуха ниже оптимальных величин, не более	для диапазона температур воздуха выше оптимальных величин, не более
Холодный	20,0 - 21,9	24,1 - 25,0	19,0 - 26,0	15 – 75	0,1	0,1
Теплый	21,0 - 22,9	25,1 - 28,0	20,0 - 29,0	15 – 75	0,1	0,2

Значения показателей, полученные при измерении на рабочем месте:

- температура воздуха 23,5 °С – оптимальное значение;
- температура поверхностей 22 °С – оптимальное значение;
- относительная влажность воздуха 65% – допустимое значение;
- скорость движения воздуха 0,1 м/с – оптимальное значение.

Все измеренные показатели удовлетворяют санитарным нормам для рабочих помещений.

Методы минимизации фактора отклонений показателей микроклимата: установка системы вентиляции, системы кондиционирования, системы отопления.

5.3.3. Недостаточная освещенность рабочей зоны

Низкая освещенность рабочей зоны губительно влияет на органы зрения работников, снижает зрительную работоспособность, а также влияет на настроение и общее самочувствие работников, определяет эффективность выполнения работы. Нерациональная организация освещения является одной из причин травматизма на рабочем месте, так как ухудшение видимости объектов и неадекватное восприятие наблюдаемых предметов может быть

спровоцировано плохо освещенными опасными зонами, слепящими источниками света, световыми бликами, резкими тенями, а также пульсацией световых источников.

В помещениях для работы с персональными компьютерами должно быть естественное и искусственное освещение. Нормативные показатели освещения в соответствии с СанПиН 2.2.1/2.1.1.1278-03 представлены в таблице 13 [20].

Таблица 13. Нормируемые показатели естественного, искусственного и совмещенного освещения (СанПиН 2.2.1/2.1.1.1278-03)

Помещения	Рабочая поверхность и плоскость нормирования КЕО и освещенности и высота плоскости над полом, м	Естественное освещение		Совмещенное освещение		Искусственное освещение				
		КЕО ен, %		КЕО ен, %		освещенность, лк			показатель дискомфота М, не более	коэффициент пульсации освещенности, Кп, % не более
		при верхнем или комбинированном освещении	при боковом освещении	при верхнем или комбинированном освещении	при боковом освещении	при комбинированном освещении		при общем освещении		
						всего	от общего			
Помещения для работы с дисплеями залы ЭВМ	Г-0,8 Экран монитора: В-1,2	3,5 -	1,2 -	2,1 -	0,7 -	500 -	300 -	400 200	15 -	10 -

На представленном рабочем месте использовано сочетание естественного и искусственного освещения, то есть освещение смешанного типа.

Естественным освещением помещение обеспечивается за счет оконных проемов, освещение должно быть с левой стороны от работника. Для искусственного освещения помещений с персональными компьютерами рекомендовано применение светильников типа ЛПО 2х36. Расположение светильников рекомендуется линиями, чтобы при разных положениях ПК светильники находились параллельно линии зрения работника, с защитным углом более 40 градусов.

Рассматривая представленное рабочее место, установим, что естественное освещение в помещении осуществляется через один оконный проем размером 2х1,5 метра в наружной стене. Искусственный свет в помещении представлен 6 светильниками типа ЛПО 36, расположенными

линиями, что дает непрерывное и равномерное освещение. В каждом светильнике установлено 4 люминесцентные лампы типа ЛБ-40.

Проведем расчет освещенности рабочего места. Исходными данными являются размеры помещения 6х7х3 м, световой поток используемых ламп равен 900 лк. Стены и потолок в помещении имеют отделку белого цвета, пол серого цвета, следовательно, индексы отражения для потолка и стен равны 80, для пола – 30.

Так как должность разработчика предполагает длительные монотонные операции с высоким уровнем зрительной работы, то есть различение объектов, размером от 3 до 5 мм, такая работа класса III, необходимо принять за норму освещенности рабочего места от 300 до 500 лк.

Коэффициент запаса, показывающий поправку на запыленность источников освещения, примем 1,2, так как запыленность значительно меньше 1 мг/м^3.

Определяем индекс помещения по формуле:

$$И_П = \frac{S}{((h_1 - h_2) \times (a + b)}$$

где $И_П$ – индекс помещения; S – площадь; h_1 – высота потолков; h_2 – высота рабочего стола; a – длина помещения; b – ширина помещения.

Для представленного рабочего места рассчитаем:

$$И_П = \frac{42}{((3 - 0{,}8) \times (6 + 7)} = 1{,}468$$

По полученному индексу помещения определим, что коэффициент использования помещения U равен 83.

Проведем расчет освещенности по следующей формуле:

$$E = \frac{К_{св} \times К_л \times СП_л \times U}{S \times k_з \times 100}$$

где $К_{СВ}$ – количество светильников; $К_Л$ – количество лампочек в светильнике; $СП_Л$ – световой поток лампочки; U – коэффициент использования; S – площадь; $k_З$ – коэффициент запаса.

Для представленного рабочего места получим:

$$E = \frac{6 \times 4 \times 900 \times 83}{30 \times 1{,}2 \times 100} = 498$$

Получено значение освещенности в 498 лк, следовательно, освещение рабочего места соответствует нормативным значениям.

Методы минимизации фактора недостаточной освещенности: установка рассеянного потолочного освещения, расстановка настольных ламп, обеспечение единой цветовой температуры всем искусственным источникам света.

5.3.4. Превышение уровня шума

Шумом называется совокупность звуков, возникающих в процессе работы и негативно воздействующих на работника, так как под воздействием шума нарушается физиологические функции, уменьшается концентрация внимания, проявляется усталость и напряжение. Таким образом, шум уменьшает работоспособность и снижает производительность работника.

Для различных категорий рабочих помещений нормативные уровни шума регламентируются ГОСТ 12.1.003-83. Помещения для работы с ПК не могут граничить с помещениями с повышенным уровнем шума. При выполнении работы на ПК уровень шума должен быть ниже 50 дБА. Оборудование, превышающее нормативный уровень шума должны находиться вне помещения для работы с ПК.

В представленном рабочем помещении основными источниками шума являются персональные компьютеры, оргтехника и кондиционер. С учетом

максимального числа работников в смену уровень шума равен 52 дБА, что немного превышает нормативное значение.

Методы минимизации фактора превышения уровня шума: использование звукопоглощающих материалов для обшивки стен, использование звукопоглощающих экранов и перегородок, очистка внутренних вентиляторов оборудования от пыли, использование берушей, герметизация щелей в помещении.

5.3.5. Повышенное значение напряжения в электрической цепи, замыкание которой может произойти через тело человека

Источниками электрической опасности являются электрические сети, оборудование, инструменты и техника. В связи с большим количеством электрических приборов и вычислительных машин на представленном рабочем месте, электробезопасность является важной составляющей производственной безопасности.

При повышенном значении напряжения в электрической цепи может произойти удар током, что приведет к термическому, биологическому и электролитическому воздействию на организм человека.

При работе с электрифицированными приборами необходимо соблюдать технику безопасности, которая представляет собой систему мероприятий и технических средств, направленных на предотвращение воздействий на работников вредных и опасных факторов.

В рабочем помещении может происходить накопление статического электричества, его разряды не представляют опасности для работников, но могут служить источником возникновения проблем с вычислительными машинами. Чтобы снизить уровень статического электричества, полы в помещении покрываются однослойным линолеумом.

Опасность поражения электрическим током являются серьезной потенциальной проблемой, так как человеческие органы чувств не могут обнаружить наличие электрического напряжения на расстоянии.

Риск поражения электрическим током возрастает при следующих условиях: повышенная влажность, когда относительная влажность воздуха выше 75 %; высокая температура воздуха и поверхностей, более 35 °С; наличие токопроводящей пыли и токопроводящих полов; возможность одновременного соприкосновения к заземленным металлическим элементам и металлическим корпусом электрооборудования.

Работа может проводиться исключительно в помещениях, исключающих повышенную опасность, однако, есть риск возникновения опасности другого рода:

- при прикосновении к токоведущим частям (во время ремонта ПК);
- при прикосновении к нетоковедущим частям, которые оказались под напряжением (при нарушении изоляции);
- при соприкосновении с полом или стенами, оказавшимися под напряжением (при нарушении электрической сети);
- при коротком замыкании в высоковольтных блоках.

Представленное место работы не относится к помещениям повышенной опасности электропоражения. В помещении используются приборы, потребляющие напряжение 220 В переменного тока с частотой 50 Гц. Для предотвращения возникновения опасных ситуаций обязательны следующие меры предосторожности:

- перед началом рабочей смены необходимо убедиться, что выключатели и розетки закреплены и не имеют оголенных токоведущих частей;
- при обнаружении неисправности электрооборудования сообщить ответственному лицу, не делая никаких самостоятельных исправлений;
- запрещено загромождать рабочее место лишними предметами.

Методы минимизации воздействия фактора повышенного значения напряжения в электрической цепи: использование устройств защиты от скачков напряжения, соблюдение правил техники безопасности, использование заземления, обновление электропроводки.

5.4. Обоснование мероприятий по снижению уровней воздействия опасных и вредных факторов на исследователя (работающего)

Для минимизации вышеописанных негативных факторов необходимо провести следующие мероприятия:

1. Применять специальные приборы-минимизаторы электромагнитного излучения;
2. Установить системы вентиляции, системы кондиционирования, системы отопления;
3. Установить рассеянное потолочное освещение, использовать настольные лампы;
4. Использовать звукопоглощающие перегородки и очистить внутренние вентиляторы оборудования от пыли;
5. Использовать устройства защиты от скачков напряжения и соблюдать правила техники безопасности;
6. Проводить медосмотр перед трудоустройством рабочего;
7. Проводить инструктаж техники безопасности;
8. Поверять электрооборудование на факт наличия оголенных электрических элементов и целостность проводки.

5.5. Экологическая безопасность

Человечество всегда оказывает влияние на окружающую среду, на текущий момент данное влияние имеет катастрофические масштабы, так как

воздействие человека и его потребление ресурсов перешло на тот уровень, когда планета не способна воспроизвести столько ресурсов, сколько потребляет человечество. Такое отношение к Земле привело к дефициту экосистем и экологическому кризису.

Наука не стоит на месте, развивается и представляет новые способы предотвращения и исправления экологических проблем. Защита окружающей среды требует полного перехода к безотходным и малоотходным производствам и технологиям, к правильной утилизации отходов. Для этого необходим комплекс технологических и организационных мероприятий, основанных на использовании современных научных достижений.

Утилизация компьютерной и организационной техники ограничено законодательно, так как в производстве такой техники используется большое количество материалов, способных нанести большой вред окружающей среде. Утилизация компьютерного оборудования происходит через обязательное извлечение компонент, их сортировку и последующую отправку для повторного использования. Такая утилизация обязательно производится на оборудованных полигонах с привлечением квалифицированного персонала.

Люминесцентные лампы являются одним из самых распространенным источником загрязнения ртутью, так как при неправильной утилизации ламп ртуть, находящаяся в них, попадает в землю, что очень опасно для планеты и для жизни людей. Правильной утилизацией люминесцентных ламп является передача лицензированным компаниям для переработки и вторичного использования сырья в качестве материала для производств.

Утилизация мусорных отходов, таких как бумажная макулатура, канцелярские принадлежности, средства личной гигиены и продукты питания, производится через сбор, обязательную сортировку и утилизацию. Отходы, которые можно использовать повторно, например, макулатуру, после сортировки отправляют на переработку через компании, занимающиеся сбором таких отходов.

Используя такую систему утилизации отходов работы можно значительно уменьшить негативное воздействие на окружающую среду, а также на собственное здоровье, поскольку качественная утилизация отходов исключает отравление опасными веществами и попадание тяжелых металлов в организмы.

5.6. Безопасность в чрезвычайных ситуациях

Чрезвычайной ситуацией (ЧС) называется обстановка на определенной территории, сложившаяся в результате аварии, опасного природного явления, катастрофы или другого бедствия, которая может привести к человеческим жертвам, причинить ущерб здоровью людей или окружающей среде, материальные потери. ЧС для представленного рабочего помещения является пожар. Данная ЧС может произойти в случае несоблюдения мер пожаробезопасности, нарушения техники использования электрических приборов и ПК, нарушениях разводки электрических сетей и ряда других причин.

Рабочее помещение, представленное для выполнения ВКР, согласно СанПиН 2.2.1/2.1.1.1278-03, можно отнести к категории В (пожароопасное).

В качестве возможных причин возникновения пожара можно указать следующие причины:

- короткое замыкание;
- опасная перегрузка сетей, которая ведет за собой сильный нагрев токоведущих частей и загорание изоляции;
- пуск оборудования после некорректного и неквалифицированного ремонта.

Для предотвращения ЧС необходимо соблюдать правила пожарной безопасности, чтобы обеспечить состояние защищенности работников и имущества от пожара.

Для защиты от коротких замыканий и перегрузок необходимо правильно выбирать, устанавливать и использовать электрические сети и средства автоматизации.

Для предупреждения возникновения пожаров необходимо исключить образование горючей среды, следить за применением при строительстве и отделке зданий несгораемых или трудно сгораемых материалов.

Необходимо проводить следующие пожарно-профилактические мероприятия:

- организационные мероприятия, касающиеся технического процесса с учетом пожарной безопасности объекта (инструктаж персонала, обучение правилам техники безопасности, издание инструкций, плакатов, планов эвакуации);
- эксплуатационные мероприятия, рассматривающие эксплуатацию используемого оборудования (соблюдение эксплуатационных норм оборудования, обеспечение свободного подхода к оборудованию, поддержание исправности изоляции проводников);
- технические и конструктивные мероприятия, связанные с правильным размещением и монтажом электрооборудования и отопительных приборов (соблюдение противопожарных мероприятий при устройстве электропроводок, оборудования, систем отопления, вентиляции и освещения).

Для повышения устойчивости рабочего помещения к ЧС необходимо произвести установку систем противопожарной сигнализации, реагирующих на дым и другие продукты горения, установку огнетушителей. Также, два раза в год проводить учебные тревоги для отработки действий при пожаре.

В представленном рабочем помещении при входе представлен план эвакуации, установлена система противопожарной сигнализации. Помещение оборудовано углекислотными огнетушителями типа ОУ-2 в количестве 2 штук на одну рабочую зону. В зоне досягаемости работниками находится

электрощит, с помощью которого можно полностью обесточить рабочее помещение.

В случае возникновения возгорания, необходимо вызвать пожарную службу по телефону 112 и сообщить место возникновения ЧС, предпринять меры по эвакуации работников в соответствии с планом эвакуации. При отсутствии прямых угроз здоровью и жизни произвести попытку тушения возникшего возгорания имеющимися углекислотными огнетушителями. В случае потери контроля над пожаром, необходимо эвакуироваться согласно плану эвакуации и ждать приезда специалистов пожарной службы.

5.7. Выводы

В заключение раздела можно сделать вывод о том, что грубых нарушений по организации работы при выполнении ВКР не обнаружено, все требования и нормы безопасности соблюдены. Организационные вопросы по обеспечению необходимых рабочих условий имеют под собой законодательное подтверждение и не нарушают законодательный регламент.

Заключение

При выполнении выпускной квалификационной работы была проведена разработка программной системы для сбора неперсональных пользовательских данных из социальной сети «Вконтакте» посредством API-запросов. Реализованные функциональные возможности были протестированы и доказали свою работоспособность.

В ходе выполнения работы было проведено исследование предметной области, которое включает в себя описание предметной области, моделирование бизнес-процессов в нотации IDEF0, обзор и анализ существующих аналогов систем управления проектами. По результатам проведенного анализа было принято решение о разработке собственной программной системы.

Перед началом реализации было проведено проектирование и функциональное моделирование будущей программной системы с использованием нотаций UML, IDEF3 и DFD.

В ходе работы были написаны все необходимые компоненты системы. Код основных компонент системы, таких как main.pas, excel.pas, uMyThread_Ids.pas, uMyThread_Members, XSuperJSON.pas, XSuperObject.pas, представлен в приложениях к выпускной квалификационной работе А, Б, В, Г, Д и Е соответственно. Интерфейс программы и примеры файлов с данными представлены в разделе «Тестирование работы программной системы».

Разработанная программная система была протестирована на реальных данных из группы «ТПУ | Томский политехнический университет», размещенной в социальной сети «Вконтакте». Собранные данные соответствуют действительным. Программа работает без сбоев и не собирает персональную информацию пользователей. Внедрение разработки позволит повысить качество социальных и маркетинговых исследований, а также позволит анализировать структуру и интересы пользовательской аудитории.

Выполнены задания по разделам «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение» и «Социальная ответственность», показавшие реальность внедрения разрабатываемого программного продукта и их актуальность.

В результате выполнения выпускной квалификационной работы были закреплены и углублены теоретические знания, получены практические навыки сбора и анализа информации, проектирования и моделирования информационных систем, программирования и тестирования разработанной программной системы.

Полученные навыки удовлетворяют описанным ранее планируемым результатам обучения по профилю специальности «Программная инженерия».

Список достижений

Статьи:

1. Осмоналиев Т. М., Савельев А. О. API социальных сетей как инструмент формирования и корректировки молодежной политики вуза: материалы докладов, XIV Международная научно-практическая конференция, 28–30 ноября 2018 г. – Томск: В-Спектр, 2018 – С. 108-112.

2. Осмоналиев Т. М. Моделирование в автомобильном строении [Электронный ресурс] // Информационные технологии в науке, управлении, социальной сфере и медицине: сборник научных трудов II Международной конференции, Томск, 19-22 мая 2015. – Томск: ТПУ, 2015 – С. 76-78. – Режим доступа: http://www.lib.tpu.ru/fulltext/c/2015/C24/C24.pdf

3. Осмоналиев Т. М., Кесельман М. М., Панькова Н. М. Актуальность идей Ф. Кафки в современном мире // сборник научных трудов, XXIX Студенческая международная заочная научно-практическая конференция «Молодежный научный форум: гуманитарные науки». – 2015 – С. 37-41.

Список использованных источников

1. Вконтакте. Правовая информация [Электронный ресурс] – Режим доступа: https://vk.com/legal, свободный (дата обращения: 07.05.2019 г.).
2. Вконтакте. Выполнение запросов к API Вконтакте [Электронный ресурс] – Режим доступа: https://vk.com/dev/api_requests, свободный (дата обращения: 29.04.2019 г.)
3. Вконтакте. Метод API users.get [Электронный ресурс] – Режим доступа https://vk.com/dev/users.get, свободный (дата обращения: 25.04.2019 г.).
4. Вконтакте. Метод API groups.getMembers [Электронный ресурс] – Режим доступа: https://vk.com/dev/groups.getMembers, свободный (дата обращения: 25.04.2019 г.).
5. КонсультантПлюс. РФ от 27 июля 2006 года №152-ФЗ «О персональных данных» [Электронный ресурс] – Режим доступа: http://www.consultant.ru/document/cons_doc_LAW_61801, свободный (дата обращения: 25.04.2019 г.).
6. Performance Lab. Деперсонализация данных [Электронный ресурс] – Режим доступа: http://www.performance-lab.ru/wp-content/uploads/2018/01/pl-depersonalization-data.pdf, свободный (дата обращения: 07.05.2019).
7. Бизнес-студио. Нотация IDEF0 [Электронный ресурс] – Режим доступа: https://www.businessstudio.ru/wiki/docs/current/doku.php/ru/csdesign/bpmodeling/idef0, свободный (дата обращения: 07.05.2019).
8. Упрвление производством. Диаграмма Исикавы [Электронный ресурс] – Режим доступа: http://www.up-pro.ru/encyclopedia/diagramma-isikavy.html, свободный (дата обращения: 07.05.2019).

9. Agora Pulse. Social Media Management Simplified [Электронный ресурс] – Режим доступа: https://agorapulse.com, свободный (дата обращения: 25.05.2019 г.).
10. YouScan. Система мониторинга социальных медиа и социальных сетей [Электронный ресурс] – Режим доступа: https://youscan.io, свободный (дата обращения: 25.05.2019 г.).
11. Embarcadero. Delphi [Электронный ресурс] – Режим доступа: https://www.embarcadero.com/ru/products/delphi, свободный (дата обращения: 26.05.2019 г.).
12. Javarush. UML-диаграммы [Электронный ресурс] – Режим доступа: https://javarush.ru/groups/posts/uml-v-java, свободный (дата обращения: 26.05.2019 г.).
13. Business Consulting Group. Диаграммы DFD [Электронный ресурс] – Режим доступа: http://b-c-group.ru/?page_id=103, свободный (дата обращения: 26.05.2019 г.).
14. Бизнес-студио. Событийная цепочка процессов [Электронный ресурс] – Режим доступа: http://www.businessstudio.com.ua/bp/bs/overview/notation_epc.php, свободный (дата обращения: 26.05.2019 г.).
15. Elma-bpm. Ввод в нотацию BPMN [Электронный ресурс] – Режим доступа: https://www.elma-bpm.ru/journal/index.php?ELEMENT_ID=2894, свободный (дата обращения: 27.05.2019 г.).
16. Вконтакте. ТПУ | Томский политехнический университет [Электронный ресурс] – Режим доступа: https://vk.com/tpunews, свободный (дата обращения: 27.05.2019 г.).
17. МегаОбучалка. Анализ конкурентных технических решений с позиции ресурсоэффективности и ресурсосбережения [Электронный ресурс] – Режим доступа: https://megaobuchalka.ru/3/25625.html, свободный (дата обращения: 27.05.2019 г.).

18. PowerBranding. Многоугольник конкурентоспособности [Электронный ресурс] – Режим доступа: http://powerbranding.ru/competition/mnogougolnik-konkurentosposobnosti, свободный (дата обращения: 27.05.2019 г.).
19. Executive. SWOT-анализ [Электронный ресурс] – Режим доступа: https://www.e-xecutive.ru/wiki/index.php/SWOT-анализ, свободный (дата обращения: 27.05.2019 г.).
20. СанПиН 2.2.1/2.1.1.1278-03. Гигиенические требования к естественному, искусственному и совмещенному освещению жилых и общественных зданий. – М.: Информационно-издательский центр Минздрава России, 2003.
21. СанПиН 2.2.4.1191-03. Электромагнитные поля в производственных условиях. – М.: Информационно-издательский центр Минздрава России, 2003.
22. СанПиН 2.2.4.1340-03. Гигиенические требования к персональным электронно-вычислительным машинам и организации работы. – М.: Информационно-издательский центр Минздрава России, 2003.
23. СанПиН 2.2.4.548-96 Гигиенические требования к микроклимату производственных помещений. – М.: Информационно-издательский центр Минздрава России, 1997.
24. Электронный фонд правовой и нормативно-технической документации. Рабочее место при выполнении работ сидя [Электронный ресурс] – Режим доступа: http://docs.cntd.ru/document/1200003913, свободный (дата обращения: 28.05.2019 г.).
25. СанПиН 2.2.2/2.4.1340-03 Гигиенические требования к персональным электронно-вычислительным машинам и организации работы: с изменениями от 3 сентября 2010 г. – М.: Информационно-издательский центр Минздрава России, 2003.

26. СанПиН 2.2.4/2.1.8.055-96. Электромагнитные излучения радиочастотного диапазона (ЭМИ РЧ). Санитарные правила и нормы от 8 мая 1996 г. – М.: Информационно-издательский центр Минздрава России, 2003.
27. СанПиН 2.2.4.548-96 Гигиенические требования к микроклимату производственных помещений: с изменениями от 1 октября 1996 г. – М.: Информационно-издательский центр Минздрава России, 2003.
28. СанПиН 2.2.1/2.1.1.1278-03 Гигиенические требования к естественному, искусственному и совмещенному освещению жилых и общественных зданий: с изменениями от 8 апреля 2003 г. – М.: Информационно-издательский центр Минздрава России, 2003.
29. СН 2.2.4/2.1.8.562-96 Шум на рабочих местах, в помещениях жилых, общественных зданий и на территории жилой застройки: с изменениями от 31 октября 1996 г. – М.: Информационно-издательский центр Минздрава России, 2003.
30. ГОСТ Р 12.1.019-2009 Система стандартов безопасности труда (ССБТ). Электробезопасность. Общие требования и номенклатура видов защиты: с изменениями от 01 сентября 2001 г. – М.: Информационно-издательский центр Минздрава России, 2003.

Приложение А. Компонент системы Main.pas

```
unit main;

interface

uses
  Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,
System.Classes, Vcl.Graphics, Vcl.Controls,
   [блок удален из-за соображений безопасности]_Members, uMyThread_Ids,
Vcl.Grids, ExcelXP, Excel, ComObj, ActiveX,
  Data.DB, Vcl.DBCtrls, Vcl.DBGrids, Data.Win.ADODB, Data.FMTBcd,
Data.SqlExpr;

type
  TmainForm = class(TForm)
    StartBitBtn: TBitBtn;
    StopBitBtn: TBitBtn;
    UserAgentListBox: TListBox;
    LogGroupBox: TGroupBox;
[блок удален из-за соображений безопасности]
    procedure StopBitBtnClick(Sender: TObject);
    procedure SaveLogBitBtnClick(Sender: TObject);
    [блок удален из-за соображений безопасности]
    private
    function CheckStartError: boolean;
    procedure Start;
    procedure InitialVariables;
    procedure Print(str: string; col: tcolor = clblack);
    procedure StartThreads_Members;
    procedure CloseAllThreads;
    procedure PauseMyThreads;
    procedure UnPauseMyThreads;
    procedure MakePauseThreads;
    procedure ScrollLog;
    procedure GetGroupMembers;
    procedure AddMemberToLB;
    procedure ShowStatistic;
    procedure BeforeStart;
    procedure AfterEnd;
    procedure LoadMembers;
    procedure SaveMembers;
    procedure SetGridHeaders;
    procedure GetIDsInfo;
    procedure StartThreads_IDs;
    procedure SaveToExcel;
    function SaveAsExcelFileNew(stringGrid: TStringGrid; FileName:
string): Boolean;
    procedure SaveToBase;
    procedure InitBase;
    procedure ClearBase;
    procedure ClearMembersTable;


    { Private declarations }
```

```
  public
    { Public declarations }
  end;

const
  PROGRAM_NAME = 'Сбор данных из Вконтакте';
  PROGRAM_VERSION = '0.1';

var
  mainForm: TmainForm;
  isStopPressed, isPaused: boolean;
[блок удален из-за соображений безопасности]
        SQL.Text := 'DELETE * FROM main';
    Prepared := True;
    ExecSQL;
  end;
end;

procedure TmainForm.InitBase;
begin
  ClearBase;

  ADOQuery1.SQL.Clear;
  ADOQuery1.SQL.Add('SELECT * FROM main');
  ADOQuery1.Active := True;
end;

procedure TmainForm.InitialVariables();
var
  fn: string;
begin
  fn := ExtractFilePath(ParamStr(0)) + 'System\agents.txt';
  if FileExists(fn) then
[блок удален из-за соображений безопасности]
      fn, header: string;
  i: integer;
  SL: TStringList;
begin
  fn := ExtractFilePath(ParamStr(0)) + 'System\headers.txt';
  if not FileExists(fn) then
  begin
    Beep;
[блок удален из-за соображений безопасности]

  Application.ProcessMessages;

  SL.Free;
end;

procedure TmainForm.FormCreate(Sender: TObject);
var
  fn: string;
begin
  Randomize; [блок удален из-за соображений безопасности]
        closefile(f);
```

```
  end;
  SD.Free
end;

procedure TmainForm.SaveMembers_BitBtnClick(Sender: TObject);
begin
  SaveMembers;
end;

procedure TmainForm.SaveReady_BtnClick(Sender: TObject);
begin
  if SG.RowCount = 0 then
    exit;
[блок удален из-за соображений безопасности]
      Application.ProcessMessages;

  SaveToBase;

  SaveToBase_Btn.Enabled := true;
  Screen.Cursor := crDefault;
[блок удален из-за соображений безопасности]
                ShellExecute(Self.Handle, 'open', PChar(SD.FileName),
nil, nil, SW_ShowNormal);
          end;
        IDNO:
          begin
           //
          end;
      end;

    end;
  end;
  SD.FRee;
end;

function TmainForm.SaveAsExcelFileNew(stringGrid: TStringGrid; FileName:
string): Boolean;
const
  xlWBATWorksheet = -4167;
var
  Row, col: Integer;
  GridPrevFile: string;
  XLApp: OLEVariant;
  i, j: Integer;
  FData: Variant;
  Sheet, Range: Variant;
  WorkBook: Variant;
  WorkSheet: Variant;
begin
  KillTask('Excel.exe');

  Result := false;
  FData := VarArrayCreate([1, stringGrid.RowCount, 1, [блок удален из-за
соображений безопасности]
      SaveReady_Btn.Enabled := false;
```

```
  isStopPressed := false;
  isPaused := false;
  StopBitBtn.Enabled := true;
[блок удален из-за соображений безопасности]
      else
    result := false;
end;

procedure TmainForm.ClearLogBitBtnClick(Sender: TObject);
begin
  Log.Clear;
  Application.ProcessMessages;
end;

procedure TmainForm.ClearMembersTable;
var
  i: integer;
begin
  SG.RowCount := 1;

  try
    SG.HandleNeeded;
    SG.Row := SG.RowCount - 1;
[блок удален из-за соображений безопасности]
    end;

procedure TmainForm.StopBitBtnClick(Sender: TObject);
begin
  StartBitBtn.Enabled := True;
  StopBitBtn.Enabled := false;
  PauseBitBtn.Enabled := false;

  Print('Нажали Стоп!', clblue);
  isStopPressed := True;
[блок удален из-за соображений безопасности]
              MyThread_IDs[i].Suspend;
    except
    end;
  end;

end;

procedure TmainForm.UnPauseMyThreads();
var
  i: Integer;
begin

  for i := 0 to 1 - 1 do
  begin
    try
[блок удален из-за соображений безопасности]
      end;

  for i := 0 to 1 - 1 do
```

```
  begin
    try
      if Assigned(MyThread_IDs[i]) then
        if MyThread_IDs[i] <> nil then
          MyThread_IDs[i].Resume;
    except
    end;
  end;

end;

procedure TmainForm.MakePauseThreads;
var
  i: Integer;
begin
  isPaused := not isPaused;
  if isPaused then
  begin
    PauseBitBtn.Font.Size := 10;
[блок удален из-за соображений безопасности]
    ;
  ListBoxKeyDown(Sender, Key, Shift);
  Members_ListBox.Items.EndUpdate;
  ShowStatistic;
end;

procedure TmainForm.Member_EditKeyDown(Sender: TObject; var Key: Word;
Shift: TShiftState);
begin
  if Key = 13 then
    AddMemberToLB;
end;

procedure TmainForm.CloseAllThreads;
var
  i: integer;
begin
  for i := 0 to 1 - 1 do
  begin
    try
      if MyThread_Members[i].isWork then
[блок удален из-за соображений безопасности]
        MyThread_IDs[i].Priority := tpLowest;
    MyThread_IDs[i].FreeOnTerminate := true;
    MyThread_IDs[i].Resume;
  end;
end;

//////////////////////////////////// - получение списка пользователей в
группе
procedure TmainForm.GetGroupMembers();
var
  url: string;
begin
  Log.Clear;
```

```
  Application.ProcessMessages;
  if CheckStartError then
    Exit;

  Members_ListBox.Items.Clear;

  successThreadsCounter := 0;
  totalThreadCounters := 1;
 [блок удален из-за соображений безопасности]
       Exit;

  if Members_ListBox.Items.Count = 0 then
  begin
    print('Список пользователей пустой и начнется автоматический сбор!');
    GetGroupMembers;
  end;
  if isStopPressed then
    exit;

  if (Members_ListBox.Items.Count = 0) then
  begin
    beep;
    print('Не удалось получить список пользователей. Работа невозможна!',
clred);
    exit;
  end;
  ShowStatistic;
  GetIDsInfo;
  ShowStatistic;
end;

end.
```

Приложение Б. Компонент системы Excel.pas

```
unit Excel;

interface

uses Windows, Activex;

// originally from Excel97.pas
// XlFileFormat constants
type
  XlFileFormat = TOleEnum;
const
  xlAddIn = $00000012;
  xlCSV = $00000006;
  xlCSVMac = $00000016;
  xlCSVMSDOS = $00000018;
  xlCSVWindows = $00000017;
  xlDBF2 = $00000007;
[блок удален из-за соображений безопасности]
  xlWebArchive = 45;
```

```
  xlWorkbookDefault = 51;
  xlXMLSpreadsheet = 46;

implementation

end.
```

Приложение В. Компонент разработанной системы uMyThread_Ids.pas

```
unit uMyThread_Ids;

interface

uses
  Winapi.Windows, Winapi.Messages, System.Classes, SysUtils, Forms,
Vcl.StdCtrls, Vcl.ComCtrls, IdIOHandler,
  IdIOHandlerSocket, IdIOHandlerStack, Graphics, IdSSL, IdSSLOpenSSL,
IdCookieManager, IdZLibCompressorBase,
  IdCompressorZLib [блок удален из-за соображений безопасности]
    home_town, interests, last_seen,
      movies, music, occupation, online, personal, quotes, relation,
schools, sex, site, status, timezone, trending, tv,
      universities, counter_friends: string;
  end;

type
  TMyThread_IDs = class(TThread)
  private
    { Private declarations }
    msg: string; // вывод для print
    currentColor: TColor;
    ////////////////////
    http: TIdHTTP;
    SSL: TIdSSLIOHandlerSocketOpenSSL;
    Socks: TIdSocksInfo;
    coockies: TIdCookieManager;
    compress: TIdCompressorZLib;
    F_IDS: string; // id пользователя
    F_Fields: string; // поля профиля, по которым будет собираться
информация
    F_Vk_User: MyVkUser;
    ////////////////////////
    function AddColorText(txt: string): string;
    procedure print(str: string; col: TColor = clBlack);
    procedure PrintMsg;
    procedure Pause(p: integer; r: boolean = false);
    function MakeRandomDelay(p: integer): integer;
    function Pars(HTML, s1, s2: string): string;
    function GetUA: string;
    procedure HttpFree;
    procedure HttpInit;
    function GetPage(url: string): string;
    procedure ParseMembers(response: string);
    procedure AddIDToSG;
```

```
    function BuildRequest(): string;
    procedure GetUserIDs_ToString;
    function GetIDsInfo: boolean;
    function ReadFields: boolean;
    function GetRelation(rel: integer): string;
    function GetSex(sex: integer): string;
  protected
    procedure Execute; override;
  public
    threadNumber: integer;
    isWork: boolean;
  end;

implementation

uses
  main;

const
  MAX_GET_IDS = 500;

function TMyThread_IDs.GetPage(url: string): string;
var
  loc, response: string;
  respCode, i, counter: Integer;
begin
  result := '';
  response := '';
  counter := 0; [блок удален из-за соображений безопасности]

  except
  end;
  try
    if coockies <> nil then
      FreeAndNil(coockies);
  except
  end;
  try
    if http <> nil then
    begin
      try
        http.Disconnect;
        FreeAndNil(http);
      except
        try
          FreeAndNil(http);
        except
          print('Не удалось очистить http!');
        end;
      end;
    end;
  except
  end;
end;
```

```
function TMyThread_IDs.Pars(HTML, s1, s2: string): string;
begin
  if (pos(s1, HTML) = 0) or (pos(s2, HTML) = 0) then
  begin
    result := '';
    Exit;
  end;

  Delete(HTML, 1, pos(s1, HTML) + Length(s1) - 1);
  result := Copy(HTML, 1, pos(s2, HTML) - 1);
end;

function TMyThread_IDs.MakeRandomDelay(p: integer): integer;
var
  sign, del: integer;
begin
  if Random(2) = 1 then
    sign := -1
   [блок удален из-за соображений безопасности]
         mainForm.Log.Lines.Add(AddColorText(TimeToStr(Now) + ' ' +
msg));
    except
    end;
  except
  end;

end;

procedure TMyThread_IDs.print(str: string; col: TColor = clBlack);
begin
  msg := str;
  currentColor := col;
  Synchronize(PrintMsg);
  Sleep(100);
end;

function TMyThread_IDs.ReadFields(): boolean; //чтения списка полей
var
  fn, field: string;
  i: integer;
[блок удален из-за соображений безопасности]
    eld;
    end;
  end;
  result := true;
  SL.Free;
end;

procedure TMyThread_IDs.AddIDToSG();
var
  i: integer;
begin
  try
[блок удален из-за соображений безопасности]
         try
```

```
        k := X_Tmp['schools'].AsArray.Length - 1;
        val := X_Tmp['schools[' + k.ToString + ']."name"'].AsString;
      except
        val := '';
      end;

      F_Vk_User.schools := val;

      k := X_Tmp['sex'].AsInteger;
      val := GetSex(k);
      F_Vk_User.sex := val;

      val := X_Tmp['site'].AsString;
      F_Vk_User.site := val;

      val := X_Tmp['status'].AsString;
      F_Vk_User.status := trim(val);

      val := X_Tmp['timezone'].AsString;
      F_Vk_User.timezone := val;
[блок удален из-за соображений безопасности]
      end;
  result := relation;
end;

function TMyThread_IDs.GetSex(sex: integer): string;
var
  sexS: string;
begin
  sexS := '';
  case sex of
    1:
      sexS := 'женский';
    2:
      sexS := 'мужской';
    0:
      sexS := 'пол не указан';

  end;
  result := sexS;
end;

procedure TMyThread_IDs.GetUserIDs_ToString;
var
  count: integer;
  id: string;
begin
  F_IDS := '';
  count := 0;
  repeat
    if mainForm.Members_ListBox.Items.count > 0 then
    begin
      id := trim(mainForm.Members_ListBox.Items[0]);
      try
[блок удален из-за соображений безопасности]
```

```
  result := url;
end;

function TMyThread_IDs.GetIDsInfo(): boolean;
var
  url, response, apiToken, group, err: string;
  param_post: TStringList;
  i: integer;
begin
  result := false;

  apiToken := trim(mainForm.ApiVk_Edit.Text);
  group := trim(mainForm.Group_Edit.Text);
  i := LastDelimiter('/', group);
  group := Copy(group, i + 1, Length(group) - 1);


[блок удален из-за соображений безопасности]
          if mainForm.Members_ListBox.Items.Count <> 0 then
        Pause(mainForm.PauseRequests_SE.Value * 1000);
      if isStopPressed then
        break;
    end;
    HttpFree;
  end;

  print('Завершен!');
  cs1.Enter;
  inc(successThreadsCounter);
  cs1.Leave;
  isWork := false;
  Terminate;
end;

end.
```

Приложение Г. Компонент системы uMyThread_Members.pas

```
unit uMyThread_Members;

interface

uses
  Winapi.Windows, Winapi.Messages, System.Classes, DateUtils, SysUtils,
Forms, Vcl.StdCtrls, Vcl.ComCtrls, IdIOHandler,
  IdIOHandlerSocket, IdIOHandlerStack, Graphics, IdSSL, IdSSLOpenSSL,
[блок удален из-за соображений безопасности]
            break;
    except
      on e: EIdException do
      begin
        Print('Ошибка открытия: ' + url);
```

```
        print('Response code: ' + http.Response.ResponseCode.ToString);
        result := 'result=ERROR';
      end;

    end;
    inc(counter);
    if counter >= 2 then
      break; // если подряд 2 ошибки к сайту, то выйти
    Pause(counter * 2000);
    print('Попытка получения страницы №' + (counter + 1).toString);
  until false;
  result := response;
end;

function TMyThread_Members.GetUA(): string;
var
  ua: string;
  i: integer;
begin
  ua := 'Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/74.0.3729.131 Safari/537.36';
  if mainForm.UserAgentListBox.Items.count > 0 then
  begin
    i := Random(mainForm.UserAgentListBox.Items.count);
    ua := mainForm.UserAgentListBox.Items[i];
   [блок удален из-за соображений безопасности]

function TMyThread_Members.Pars(HTML, s1, s2: string): string;
begin
  if (pos(s1, HTML) = 0) or (pos(s2, HTML) = 0) then
  begin
    result := '';
    Exit;
  end;

  Delete(HTML, 1, pos(s1, HTML) + Length(s1) - 1);
  result := Copy(HTML, 1, pos(s2, HTML) - 1);
end;

function TMyThread_Members.MakeRandomDelay(p: integer): integer;
var
  sign, del: integer;
begin
  if Random(2) = 1 then
    sign := -1
  else
    sign := 1;
  del := Random(p div 10);
  result := p + sign * del;
end;

procedure TMyThread_Members.Pause(p: integer; r: boolean = false);
var
  pa1: real;
  h, m, s, ms: integer;
```

```
begin
  if r then
    p := MakeRandomDelay(p);
[блок удален из-за соображений безопасности]

  count := X['response."count"'].AsInteger;
  if F_MembersCount = 0 then
    print('Количество людей в группе: ' + count.ToString);
  F_MembersCount := count;

  good := 0;
  for i := 0 to X['response."items"'].AsArray.Length - 1 do
  begin
    try
      F_ID := X['response."items"[' + i.ToString + ']'].AsString;
      inc(good);
      Synchronize(AddMemberToLB);
    except
    end;
  end;
  print('Количество собранных id пользователей: ' + good.ToString);

end;

function TMyThread_Members.BuildRequest(offset: integer): string;
var
  url, apiToken, group, response: string;
  i: integer;
begin
  apiToken := trim(mainForm.ApiVk_Edit.Text);
  group := trim(mainForm.Group_Edit.Text);
  i := LastDelimiter('/', group);
  group := Copy(group, i + 1, Length(group) - 1);
  if mainForm.Debug_CB.Checked then
    print('Работаем с группой: ' + group);

  url := 'https://api.vk.com/api.php?oauth=1&method=groups.getMembers';
  url := url + '&group_id=' + group + '&access_token=' + apiToken +
'&v=5.95';
  url := url + '&offset=' + offset.ToString + '&count=1000';

[блок удален из-за соображений безопасности]
        GetMembers(offset);

    offset := offset + 1000;
    if offset > F_MembersCount then
    begin
      print('Дошли до последней страницы сбора пользователей!');
      break;
    end;
    Pause(mainForm.PauseRequests_SE.Value * 1000);

    if isStopPressed then
      break;
  end;
```

```
  HttpFree;
  print('Завершен!');
  cs1.Enter;
  inc(successThreadsCounter);
  cs1.Leave;
  isWork := false;
  Terminate;
end;

end.
```

Приложение Д. Компонент системы XSuperJSON.pas

```
unit XSuperJSON;

interface

uses
  SysUtils, Classes, Generics.Collections, Generics.Defaults, Math,
DateUtils, RegularExpressions, RTTI;


const
  CNull = 'null';
  MaxCHR = #127;

  Err_UnexpectedEndOfInput = 'Unexpected end of input';
[блок удален из-за соображений безопасности]
      IJSONNull = interface(IJSONValue<Boolean>)['{C19F5715-B832-46D8-
8668-1A9DC31393D7}']end;
  TJSONNull = class(TJSONValue<Boolean>, IJSONNull)
  public
    procedure AsJSONString(Str: TJSONWriter); override;
  protected
    function GetIsNull: Boolean; override;
  end;

  IJSONBoolean = interface(IJSONValue<Boolean>)['{CCC8D8C5-081D-4DCF-
93DB-CC0696458A12}']end;
  TJSONBoolean = class(TJSONValue<Boolean>, IJSONBoolean)
  public
    procedure AsJSONString(Str: TJSONWriter); override;
    property Value;
  end;

   [блок удален из-за соображений безопасности]

  TJSONPair = class(TInterfacedObject, IJSONPair)
  private
    FName: String;
    FValue: IJSONAncestor;
    function GetName: String;
    function GetValue: IJSONAncestor;
```

```
    procedure SetName(const Value: String);
    procedure SetValue(const Value: IJSONAncestor);
  public
    constructor Create(const aName: String; aValue: IJSONAncestor);
    destructor Destroy; override;
    property Name: String read GetName write SetName;
    property JSONValue: IJSONAncestor read GetValue write SetValue;
  end;
[блок удален из-за соображений безопасности]
        ED: Boolean;
    BK: Boolean;
    constructor Create(NextRoute: TRoute; TriggerProcs: TTriggerProcs;
ParseProcs: TParseProc);
  end;

  TErrorTrigger = class(TTrigger)
  private
    FMessage: String;
    function GetMeessage: String;
    procedure SetMessage(const Value: String);
  public
    constructor Create(const Message: String);
    property Message: String read GetMeessage write SetMessage;
  end;


  TNoRouteTrigger = class(TTrigger)
  end;

  TUseRouteTrigger = class(TTrigger)
  end;

  TJumpTrigger = class(TTrigger)
  end;

  {$WARNINGS OFF}
  TRouteChars = set of Char;
  {$WARNINGS ON}

  TRoute = class
  private
    FName: String;
    FTriggers: array[#0..MaxCHR] of TTrigger;
    FTriggerList: TObjectList<TTrigger>;
    function GetIndex(Ch: WideChar): TTrigger; inline;
    function GetName: String;
  public
    constructor Create(const Name: String);
    destructor Destroy; override;
    property Name: String read GetName;
    procedure Add(const Chars: TRouteChars; Trigger: TTrigger);
    procedure NoRoute(Trigger: TTrigger);
    function TryGetRoute(Ch: WideChar; var Trg: TTrigger): Boolean;
inline;
    property Index[Ch: WideChar]: TTrigger read GetIndex; default;
```

```
  end;

  TLexGrammar = class
  private
     FRoutes: TList<TRoute>;
  protected
[блок удален из-за соображений безопасности]

const
  FloatFormat : TFormatSettings = ( DecimalSeparator : '.' );
  STokenTypes : array [TLexemType] of string = ('Nil',
                'String', 'Integer', 'Float', 'Null', '[', ']',
                '(', ')', '\', ':', '.', ',',
                '',
                'TRUE',
                'FALSE' );


  optAll = [#0..#255];
  optWhiteSpace = [' ', #0, #9, #10, #13];

  optAlpha = ['a'..'z', 'A'..'Z', '$', '_', #127];
  optSym = ['[', ']', '{', '}', ':', ',', '"', '''', '.'];
   [блок удален из-за соображений безопасности]

       Ord('0')..Ord('9') :
          J := J - Ord('0');
    else
       Continue;
    end;
    Result := (Result shl 4) or J;
  end;
end;

function TLexBuff.AsInt64: Int64;
begin
  Result := StrToInt64(AsString);
end;

function TLexBuff.AsString: String;
begin
  SetString(Result, Buff, Length);
end;

function TLexBuff.AsType: TLexemType;
begin
  Result := ltName;
   if Length = 0 then
      Exit;

   case Buff[0] of
      '[': Result := ltCLeft;
      ']': Result := ltCRight;
      ':': Result := ltColon;
      ',': Result := ltVirgule;
```

```
      '{': Result := ltBLeft;
      '}': Result := ltBRight;
      '.': Result := ltDot;
   else
      if CompareText(STokenTypes[ltTrue], AsString) = 0 then
         Result := ltTrue
      else
      if CompareText(STokenTypes[ltFalse], AsString) = 0 then
         Result := ltFalse
      else
      if CompareText(STokenTypes[ltNull], AsString) = 0 then
         Result := ltNull
   end;
end;

procedure TLexBuff.Clear;
begin
  Length := 0;
[блок удален из-за соображений безопасности]
    end;

function TRoute.TryGetRoute(Ch: WideChar; var Trg: TTrigger): Boolean;
begin
  if Ch > MaxCHR then Ch := MaxCHR;
  if FTriggers[Ch] <> nil then
  begin
     Result := True;
     Trg := FTriggers[Ch];
  end
  else
     Result := False;
end;

{ TLexGrammar }

constructor TLexGrammar.Create;
begin
  FRoutes := TList<TRoute>.Create;
end;

destructor TLexGrammar.Destroy;
begin
  FRoutes.Free;
  inherited;
end;

function TLexGrammar.EscapeRoute: TRoute;
begin
  Result := Nil;
end;

function TLexGrammar.EscapeSupport: Boolean;
begin
  Result := False;
end;
```

```
function TLexGrammar.CreateRoute(const Name: String): TRoute;
begin
  Result := TRoute.Create(Name);
  FRoutes.Add(Result);
end;

{ TJSONGrammar }

constructor TJSONGrammar.Create;
begin
  inherited;

  rFirst := CreateRoute('First');
[блок удален из-за соображений безопасности]
    end;

procedure TErrorTrigger.SetMessage(const Value: String);
begin
  FMessage := Value;
end;

{ TLexGenerator }

function TLexGenerator.Check(LTyp: TLexemTypes): TLexemType;
begin
  if not Assigned(FLexem) then
  begin
     NextLex;
     if not Assigned(FLexem) then
        Exit(ltNil);
  end;
  Result := iff(FLexem.LType in LTyp, FLexem.LType, ltNil);
end;

function TLexGenerator.Check(LTyp: TLexemType): Boolean;
begin
  if not Assigned(FLexem) then
  begin
     NextLex;
     if not Assigned(FLexem) then
        Exit(False);
  end;
  Result := FLexem.LType = LTyp;
end;

function TLexGenerator.CheckKill(LTyp: TLexemType): Boolean;
begin
  if not Assigned(FLexem) then
  begin
     NextLex;
     if not Assigned(FLexem) then
        Exit(False);
  end;
  if FLexem.LType = LTyp then
```

```
  begin
     KillLex;
     Result := True;
  end
  else
     Result := False;
end;

function TLexGenerator.CheckKill(LTyp: TLexemTypes): TLexemType;
begin
  if not Assigned(FLexem) then
  begin
     NextLex;
     if not Assigned(FLexem) then
        Exit(ltNil);
  end;
  if FLexem.LType in LTyp then
  begin
     Result := FLexem.LType;
     KillLex;
  end
  else
     Result := ltNil;
end;

function TLexGenerator.CheckName(var S: String): Boolean;
var
  lt: TLexemType;
begin
  lt := Check([ltSValue, ltName, ltDValue, ltIValue, ltTrue, ltFalse]);
  if lt in [ltSValue, ltName, ltTrue, ltFalse] then
  begin
     if (Pos(#$D, FLexem.Str) > 0) or (Pos(#$A, FLexem.Str)>0) then
         Exit(False);
     Result := True;
     S := FLexem.Str;
  end
  else
     Result := False;
end;

constructor TLexGenerator.Create(LexG: TLexGrammar; ExceptBlock: [блок
удален из-за соображений безопасности]
           Trigger := Route[FCurr^];
       UseEscape := False;
     end;

     if Trigger = Nil then Exit;

     CTyp := Trigger.ClassType;

     if CTyp = TErrorTrigger then
        if FCurr^ = #0 then
           Break
        else
```

```
            if FExceptBlock then
               Abort
            else
               raise TJSONSyntaxError.Create(
TErrorTrigger(Trigger).Message, FCurrPos);

     if CTyp = TUseRouteTrigger then
        if UseEscape then
           FEscapeBuff.Add(FCurr^)
        else
           FBuffer.Add(FCurr^);

     if Trigger.BF and (FLexem.Pos.Col = 0) then
        FLexem.Pos := FCurrPos^;

[блок удален из-за соображений безопасности]
    procedure TJSONBuilder.ReadObject(var Val: IJSONAncestor);
var
  Name: String;
begin
  LGen.KillLex;
  Val := TJSONObject.Create;
  repeat
    if LGen.CheckName(Name) then
    begin
       LGen.KillLex;
       if not LGen.CheckKill(ltColon) then
          raise TJSONSyntaxError.CreateFmt(Err_Expected, [':'],
LGen.CurrPos);
       TJSONObject(Val).AddPair(TJSONPair.Create(Name, ReadValue));
    end
  until not LGen.CheckKill(ltVirgule);

  if not LGen.CheckKill(ltBRight) then
     raise TJSONSyntaxError.Create(Err_UnexpectedEndOfInput,
LGen.CurrPos);
end;

procedure TJSONBuilder.ReadString(var Val: IJSONAncestor);
var
  dT: TDateTime;
  DVal: TDataType;
label
  JMP;
begin
  if (not FCheckDates) or (Length(LGen.Current.Str) > 25 {2015-10-
20T12:22:24+00:00}) or (Length(LGen.Current.Str) < 5 {22:22}) then
     JMP:Val := TJSONString.Create( LGen.Current.Str )
  else
     if TJSONDateManager.Check(LGen.Current.Str, dT, DVal ) then
[блок удален из-за соображений безопасности]
           Str.Append(#$D#$A);
    Str.Append('}');
  end;
end;
```

```
function TJSONObject.Count: Integer;
begin
  Result := FPairList.Count;
end;

constructor TJSONObject.Create;
begin
  FPairList := TList<IJSONPair>.Create;
end;

destructor TJSONObject.Destroy;
begin
  FPairList.Free;
  inherited;
end;

function TJSONObject.Get(const Name: String): IJSONPair;
var
  P: IJSONPair;
[блок удален из-за соображений безопасности]
    end;

constructor TJSONValue<T>.CreateNull;
begin
  FNull := True;
end;

function TJSONValue<T>.GetData: T;
begin
  Result := FData;
end;

function TJSONValue<T>.GetIsNull: Boolean;
begin
  Result := FNull;
end;

procedure TJSONValue<T>.SetData(const Value: T);
begin
  FData := Value;
end;

procedure TJSONValue<T>.SetNull;
begin
  FNull := True;
end;

{ TJSONInterpreter }

constructor TJSONInterpreter.Create(const Expression: String;
  JSON: IJSONAncestor; BlockException: Boolean = False);
begin
  LGen := TLexGenerator.Create(JSONLexGrammar, BlockException);
  LGen.Load(Expression);
```

```
  FJSON := JSON;
  FExceptionBlock := BlockException;
end;

procedure TJSONInterpreter.CreateExcept(const S: String;
  Args: array of TVarRec);
begin
  if FExceptionBlock then
     Abort
  else
     raise TJSONSyntaxError.CreateFmt(S, Args, LGen.CurrPos);
end;

procedure TJSONInterpreter.CreateExcept(const S: String);
begin
  if FExceptionBlock then
     Abort
  else
     raise TJSONSyntaxError.Create(S, LGen.CurrPos);
end;

destructor TJSONInterpreter.Destroy;
begin
  LGen.Free;
  inherited;
end;

function TJSONInterpreter.ReadArray(Base: IJSONAncestor): IJSONArray;
var
  Item: IJSONAncestor;
begin
  LGen.KillLex;
  Result := TJSONArray.Create;
  repeat
    Item := ReadValue(Base);
    if Assigned(Item) then
       TJSONArray(Result).Add(Item);
[блок удален из-за соображений безопасности]
      Result := Index < List.Count - 1;
  if Result then
    Inc(Index);
end;

{ TJSONWriter }

function TJSONWriter.Append(const Value: string; const CRLF: Boolean =
False): TJSONWriter;
begin
  if FIdent then
  begin
     FData.Append(' ', FIdentOffset);
     if CRLF then
        FData.AppendLine(Value)
     else
        FData.Append(Value)
```

```
  end
  else
     FData.Append(Value);
  Result := Self;
end;

function TJSONWriter.Append(const Value: int64; const CRLF: Boolean):
TJSONWriter;
begin
  Result := Append(IntToStr(Value), CRLF);
end;

[блок удален из-за соображений безопасности]

function TISO8601.ReadDate: Boolean;
begin
 Result := True;
 if not TryEncodeDate( GetIntData(NextOffset), GetIntData(NextOffset),
GetIntData(NextOffset), FValue ) then
 begin
    FValue := 0;
    Result := False;
 end
 else
    FUseDate := True
end;

procedure TISO8601.ReadMS;
var
  Temp: TDateTime;
begin
  if TryEncodeTime(0, 0, 0, GetIntData(NextOffset), Temp) then
     FValue := FValue + TTime(Temp);
end;

function TISO8601.ReadTime: Boolean;
var
  Temp: TDateTime;
begin
  if TryEncodeTime(GetIntData(NextOffset), GetIntData(NextOffset),
GetIntData(NextOffset), 0, Temp ) then
  begin
     FValue := FValue + TTime(Temp);
[блок удален из-за соображений безопасности]
                Result := Success;
            if Result then
            begin
               AValue := Value;
               Typ := ValueType;
            end;
         end;
     end);

finalization
```

```
  JSONLexGrammar.Free;

end.
```

Приложение Е. Компонент системы XSuperObject.pas

```
  TJSONType = (jtObject, jtArray);

  Alias = class(TCustomAttribute)
  private
    FName: String;
  public
    constructor Create(const AName: String);
    property Name: String read FName write FName;
  end;

  TRevalOption = (roNone, roEmptyArrayToNull);

  REVAL = class(TCustomAttribute)
  private
    FOption: TRevalOption;
    FEqual: Variant;
    FValue: Variant;
  public [блок удален из-за соображений безопасности]
    SuperArray;
    function AsArray: ISuperArray; override;
    function AsObject: ISuperObject; override;
    function Where(const Cond: TCondCallBack<IMember>): ISuperArray;
    function T: TSuperArray; inline;
    function AsType<T>: T;
  end;

  TSuperProperty = class(TRttiProperty)
  public
    ArrayRawData: Pointer;
  end;

  TSuperField = class(TRttiField)
  public
    ArrayRawData: Pointer;
  end;

  TSuperDynArr = class(TRttiDynamicArrayType)
  public
[блок удален из-за соображений безопасности]
        class function  ObjectConstructorParamCount(Instance: TClass):
Integer;
    class function  ObjectConstructor(Instance: TClass): TObject;
    class function  CheckObject<Typ>(Data: Pointer; Member: TRttiObject;
MIdx: Typ; var Obj: TObject): Boolean;
```

```
    class property  GenericsCache: TObjectDictionary<TClass,
TGenericsInfo> read FGenericsCache;
  end;

  TMemberVisibilities = set of TMemberVisibility;
  TSerializeParseOptions = class
  private
    class var FVisibilities: TMemberVisibilities;
  public
    class constructor Create;
    class property Visibilities: TMemberVisibilities read FVisibilities
write FVisibilities;
  end;

  TSuperObjectHelper = class helper for TObject
  public
    function AsJSON(const Ident: Boolean = False; const UniversalTime:
Boolean = False): String;
    function AsJSONObject: ISuperObject;
    procedure AssignFromJSON(const JSON: String); overload;
    procedure AssignFromJSON(JSON: ISuperObject); overload;
    constructor FromJSON(const JSON: String); overload;
    constructor FromJSON(JSON: ISuperObject); overload;
    constructor FromJSON(const JSON: String; CreateArgs: Array of TValue;
const ConstructMethod: String = 'Create'); overload;
    constructor FromJSON(const JSON: ISuperObject; CreateArgs: Array of
TValue; const ConstructMethod: String = 'Create'); overload;
  end;

  TBaseSuperRecord<T> = class
  public
    class function AsJSON(Rec: T): String;
    class function AsJSONObject(Rec: T): ISuperObject;
    class function FromJSON(JSON: String): T; overload;
    class function FromJSON(JSON: ISuperObject): T; overload;
  end;
  TSuperRecord<T: Record> = class(TBaseSuperRecord<T>);

  TJSON = class
[блок удален из-за соображений безопасности]
      Result := TSuperObject.Create;
  Members := SA(Args);
  if Odd(Members.Length) then
     Assert(False);
  for I := 0 to (Members.Length div 2) - 1 do
      Result.Add(Members.S[I*2], Members.Ancestor[(I*2)+1]);
end;

function SA(JSON: String): ISuperArray;
begin
  Result := TSuperArray.Create(JSON);
end;

function SA(const Args: array of const): ISuperArray;
var
```

```
  I: Integer;
  SArray: ISuperArray;
  SObject: ISuperObject;
begin
  Result := TSuperArray.Create;
  for I := 0 to High(Args) do

 [блок удален из-за соображений безопасности]
          Result :=
TT(ty.GetMethod('Create').Invoke(ty.AsInstance.MetaclassType,
[]).AsObject);
    except
      if Assigned(ty) then
        ty.Free;
      raise;
    end;
    r.Free;
  end;
end;

destructor TBaseJSON<T, Typ>.Destroy;
begin
  inherited;
end;

function TBaseJSON<T, Typ>.GetBoolean(V: Typ): Boolean;
begin
  Result := False;
  if Member(V) then
    Result := GetValue<TJSONAncestor>(V).ValueEx<Boolean>;
end;

function TBaseJSON<T, Typ>.GetData(Key: Typ): IJSONAncestor;
var
  P: IJsonPair;
begin
  if Self.InheritsFrom(TSuperObject) then
  begin
     P := TJSONObject(FInterface).Get(PString(@Key)^);
     if Assigned(P) then
        Result := P.JsonValue
     else
        Result := Nil
  end
  else
  if Self.InheritsFrom(TSuperArray) then
     Result := TJSONArray(FInterface).Get(PInteger(@Key)^);
end;

function TBaseJSON<T, Typ>.GetDataType: TDataType;
var
  Cast: ICast;
begin
  if TValue.From<T>(FJSONObj).AsInterface <> nil then
     Cast := TCast.CreateFrom<T>(FJSONObj)
```

```
  else
  if Assigned(FCasted)  then
     Cast := TCast.Create(FCasted)
  else
     Exit(dtNil);
  Result := Cast.DataType
end;

function TBaseJSON<T, Typ>.GetDate(V: Typ): TDate;
begin
  Result := 0;
  if Member(V) then
     Result := GetValue<TJSONDate>(V).Value;
end;

function TBaseJSON<T, Typ>.GetDateTime(V: Typ): TDateTime;
begin
  Result  [блок удален из-за соображений безопасности]
    (Key: Typ): TVarType;
var
  Temp: IJSONAncestor;
begin
  Temp := GetData(Key);
  if Temp = Nil then
     Result := varUnknown
  else if Temp is TJSONString then
     Result := varString
  else if Temp is TJSONFloat then
     Result := varDouble
  else if Temp is TJSONInteger then
      Result := varInt64
  else if Temp is TJSONNull then
     Result := varNull
  else if Temp is TJSONObject then
     Result := varObject
  else if Temp is TJSONArray then
     Result := varArray
  else if Temp is TJSONBoolean then
     Result := varBoolean
  else if (Temp is TJSONDateTime) or (Temp is TJSONDate) or (Temp is
TJSONTime) then
     Result := varDate
end;

procedure TBaseJSON<T, Typ>.SetArray(V: Typ; const Value: ISuperArray);
begin
  Member<TJSONArray, IJSONArray>(V, Value.Self )
end;

procedure TBaseJSON<T, Typ>.SetBoolean(V: Typ; const Value: Boolean);
begin
  Member<TJSONBoolean, Boolean>(V, Value)
end;

procedure TBaseJSON<T, Typ>.SetDate(V: Typ; const Value: TDate);
```

```
begin
  Member<TJSONDate, TDate>(V, Value);
end;

procedure TBaseJSON<T, Typ>.SetDateTime(V: Typ; const Value: TDateTime);
begin
  Member<TJSONDateTime, TDateTime>(V, Value);
end;

procedure TBaseJSON<T, Typ>.SetDouble(V: Typ; const Value: Double);
begin
  Member<TJSONFloat, Double>(V, Value);
end;

procedure TBaseJSON<T, Typ>.SetInteger(V: Typ; const Value: Int64);
begin
  Member<TJSONInteger, Int64>(V, Value);
end;

procedure TBaseJSON<T, Typ>.SetNull(V: Typ; const Value: TMemberStatus);
begin
end;

procedure TBaseJSON<T, Typ>.SetObject(V: Typ; const Value: ISuperObject);
begin
  Member<TJSONObject, IJSONObject>(V, Value.Self )
end;

procedure TBaseJSON<T, Typ>.SetString(V: Typ; const Value: String);
var
  Anc: IJSONAncestor;
  dT: TDateTime;
  ValType: TDataType;
label
  JMP, JERR;
begin
  if FCheckDate then
  begin
     Anc := Ancestor[V];
     if Assigned(Anc) and (Anc.DataType in [dtDateTime..dtTime]) then
     begin
        if not TJSONDateManager.Check(Value, dT, ValType ) then
           JERR: raise SOInvalidDate.Create('Invalid date format.')
        else
           case ValType of
             dtDateTime: Member<TJSONDateTime, TDateTime>(V, dT);
             dtDate: Member<TJSONDate, TDate>(V, TDate(dT));
             dtTime: Member<TJSONTime, TTime>(V, TTime(dT));
             else
               goto JERR;
           end;
     end
     else
       goto JMP;
  end
```

```
  else
     JMP: Member<TJSONString, String>(V, Value);
end;

procedure TBaseJSON<T, Typ>.SetTime(V: Typ; const Value: TTime);
begin
  Member<TJSONTime, TTime>(V, Value);
end;

procedure TBaseJSON<T, Typ>.SetVariant(V: Typ; const Value: Variant);
var
  VTyp: TVarType;
begin
  if VarIsNull(Value) then
     Null[V] := jNull
  else
  begin
     VTyp := GetType(V);
     if VTyp = varUnknown then
        VTyp := VarType(Value);
     case VTyp of
       varString, varUString:
          S[V] := Value;
       varInt64, varInteger, varByte:
          I[V] := Value;
       varDouble, varCurrency:
          F[V] := Value;
       varBoolean:
          B[V] := Value;
       varDate:
          D[V] := Value;
       varNull:
          Null[V] := jNull;
     end;
  end;
end;

function TBaseJSON<T, Typ>.GetSelf: T;
begin
  Result := FJSONObj;
end;


{ TSuperObject }
[блок удален из-за соображений безопасности]
      Result.Index := -1;
  Result.List := TJSONObject(FJSONObj).GetEnumerator
end;

function TSuperObject.GetEoF: Boolean;
begin
  Result := FOffset > Count - 1;
end;

function TSuperObject.GetExpr(const Code: String): ISuperExpression;
```

```
begin
  Result := TSuperExpression.Create(FJSONObj, Code);
end;


function TSuperObject.GetOffset: Integer;
begin
  Result := FOffset;
end;

function TSuperObject.GetRaw(V: String): String;
begin
  Result := GetValue<TJSONRaw>(V).ValueEx<String>;
end;

function TSuperObject.GetString(V: String): String;
begin
  Result := inherited GetString(V);
end;

procedure TSuperObject.Next;
begin
  Inc(FOffset);
end;

class function TSuperObject.ParseFile(FileName: String; CheckDate:
Boolean): TSuperObject;
var
  Strm: TFileStream;
begin
  Strm := TFileStream.Create(FileName, fmOpenRead, fmShareDenyWrite);
  try
    Result := ParseStream(Strm, CheckDate);
  finally
    Strm.Free;
  end;
end;

class function TSuperObject.ParseStream(Stream: TStream; CheckDate:
Boolean): TSuperObject;
var
  Strm: TStringStream;
begin
  Strm := TStringStream.Create;
  try
    Strm.LoadFromStream(Stream);
    Result := TSuperObject.Create(Strm.DataString, CheckDate);
  finally
    Strm.Free;
  end;
end;

procedure TSuperObject.Remove(Key: String);
begin
  FJSONObj.Remove(Key);
```

```
end;

procedure TSuperObject.SaveTo(Stream: TStream; const Ident,
UniversalTime: Boolean);
var
  S: TStringStream;
begin
  S := TStringStream.Create( AsJSON(Ident, UniversalTime) );
  try
     S.SaveToStream(Stream);
  finally
     S.Free;
  end;
end;

procedure TSuperObject.SaveTo(AFile: String; const Ident, UniversalTime:
Boolean);
var
  S: TStringSt [блок удален из-за соображений безопасности]
    (Typ) then begin
          Result :=
TValue.From<TObject>(TSerializeParse.ObjectConstructor(Typ.AsInstance.Met
aclassType)).AsType<T>;
          TSerializeParse.WriteGeneric(TValue.From<T>(Result).AsObject,
Self);
          Exit;
       end else if TSerializeParse.IsCollection(Typ) then begin
          Result :=
TValue.From<TObject>(TSerializeParse.ObjectConstructor(Typ.AsInstance.Met
aclassType)).AsType<T>;
          TSerializeParse.WriteCollection(TValue.From<T>(Result).AsObject
as TCollection, Self);
          Exit;
       end;
    end;
    raise SOException.Create('Unsupported type.');
  except
    Ctx.Free;
    raise;
  end;
end;

procedure TSuperArray.Add(Value: IJSONAncestor);
begin
  TJSONArray(FJSONObj).Add(Value);
end;

procedure TSuperArray.Clear;
begin
  FJSONObj.Clear;
end;

function TSuperArray.Clone: ISuperArray;
begin
  Result := SA(AsJSON);
```

```
end;

function TSuperArray.Delete(const Cond: TCondCallBack<IMember>):
ISuperArray;
var
  Member: IJSONAncestor;
begin
  Result := Self;
  if not Assigned(Cond) then
     Exit;
  for Member in FJSONObj do
      if Cond(TCast.Create(Member)) then
         Result.Self.Remove(Member);
end;

procedure TSuperArray.Delete(Index: Integer);
begin
  TJsonArray(FJSONObj).Remove(Index);
end;

function TSuperArray.GetEnumerator: TSuperEnumerator<IJSONAncestor>;
begin
  Result.Index := -1;
  Result.List := TJSONArray(FJSONObj).GetEnumerator
end;

function TSuperArray.GetLength: Integer;
begin
  Result := TJSONArray(FJSONObj).Count;
end;

procedure TSuperArray.SaveTo(Stream: TStream; const Ident, UniversalTime:
Boolean);
var
  S: TStringStream;
begin
  S := TStringStream.Create( AsJSON(Ident, UniversalTime) );
  try
     S.SaveToStream(S);
  finally
     S.Free;
[блок удален из-за соображений безопасности]
      end;
end;

procedure TSuperObjectHelper.AssignFromJSON(const JSON: String);
begin
  TSerializeParse.WriteObject(Self, SO(JSON));
end;

procedure TSuperObjectHelper.AssignFromJSON(JSON: ISuperObject);
begin
  TSerializeParse.WriteObject(Self, JSON);
end;
```

```
constructor TSuperObjectHelper.FromJSON(const JSON: String; CreateArgs:
array of TValue; const ConstructMethod: String);
var
  IData: ISuperObject;
begin
  IData := TSuperObject.Create(JSON);
  FromJSON(IData, CreateArgs, ConstructMethod);
end;

constructor TSuperObjectHelper.FromJSON(const JSON: ISuperObject;
CreateArgs: array of TValue; const ConstructMethod: String);
var
  Ctx: TRttiContext;
  Typ: TRttiType;
  Method: TRttiMethod;
begin
  Ctx := TRttiContext.Create;
  try
    Typ := Ctx.GetType(ClassType);
    if not Assigned(Typ) then Exit;
    Method := Typ.GetMethod(ConstructMethod);
    if (not Assigned(Method)) or not Method.IsConstructor then Exit;
    Method.Invoke(Self, CreateArgs);
  finally
    Ctx.Free;
    TSerializeParse.WriteObject(Self, JSON);
  end;
end;

constructor TSuperObjectHelper.FromJSON(JSON: ISuperObject);
begin
  FromJSON(JSON, []);
end;

{ TSerializeParse }

class procedure TSerializeParse.ReadMembers(Data: Pointer; aType:
TRttiType; IJsonData: ISuperObject);
var
  Prop: TRttiProperty;
  Field: TRttiField;
  MemberName: String;
  RevalAttribute: REVAL;
  Value: TValue;
  Attributes: TArray<TCustomAttribute>;
begin
  for Prop in aType.GetProperties do
  begin
     [блок удален из-за соображений безопасности]
     TRttiField then begin
     Result := TRttiField(Member).FieldType;
     if GetArray and (TSuperField(Member).ArrayRawData <> Nil) then
        if Result is TRttiArrayType then
           Result := TRttiArrayType(Result).ElementType
        else
```

```
            Result := TRttiDynamicArrayType(Result).ElementType;

  end else if Member is TRttiDynamicArrayType then begin
     Result := TRttiDynamicArrayType(Member).ElementType

  end else if Member is TRttiArrayType then begin
     Result := TRttiArrayType(Member).ElementType

  end;
end;
{$WARNINGS ON}

class function TSerializeParse.GetMemberTypeInfo(
  Member: TRttiObject; const GetArray: Boolean): PTypeInfo;
begin
  Result := GetMemberType(Member, GetArray).Handle
end;

class function TSerializeParse.GetREVAL(const Attributues:
TArray<TCustomAttribute>): REVAL;
begin
  Result := REVAL(GetAttribute(REVAL, Attributues));
end;

class function TSerializeParse.PropGetterType(Prop: TRttiProperty):
TPropertyGetterType;
var
  Getter: Pointer;
begin
  if Prop is TRttiInstanceProperty then begin
     Getter := TRttiInstanceProperty(Prop).PropInfo^.GetProc;
     if (IntPtr(Getter) and PROPSLOT_MASK) <> PROPSLOT_FIELD then
        Exit(pgtMethod);
  end;
  Result := pgtField;
end;

class function TSerializeParse.GetValue<Typ>(Data: Pointer;
  Member: TRttiObject; MIdx: Typ): TValue;
begin
  if (TypeInfo(Typ) = TypeInfo(Integer) ) and ( GetMemberTypeInfo(Member,
False).Kind in [tkDynArray, tkArray] ) then
      Result := GetValue<String>(GetArrayRawData(Member), Member,
'').GetArrayElement(PInteger(@MIdx)^)

  else if Member is TRttiProperty  then begin
     if (TRttiProperty(Member).PropertyType.Handle.Kind = tkDynArray) and
(PropGetterType(TRttiProperty(Member)) = pgtMethod) then begin
        TValue.Make(Nil, TRttiProperty(Member).PropertyType.Handle,
Result);
        Exit;
     end;
     Result := TRttiProperty(Member).GetValue(Data)

  end else if Member is TRttiField then
```

```
     Result := TRttiField(Member).GetValue(Data)

  else if Member is TRttiDynamicArrayType then begin
     TValue.Make(GetArrayRawData(Member),
TRttiDynamicArrayType(Member).Handle, Result);
     Result := Result.GetArrayElement(PInteger(@MIdx)^);
  end;
end;

class function TSerializeParse.IsCollection(Cls: TRttiType): Boolean;
begin
  if Cls = Nil then Exit(False);
  Result := Cls.AsInstance.MetaclassType.InheritsFrom(TCollection);
end;

class function TSerializeParse.IsCollection(Cls: TClass): Boolean;
begin
  with TRttiContext.Create do
    try
      Result := IsCollection(GetType(Cls));
    finally
      Free;
    end;
end;

class function TSerializeParse.IsDisabled(const Attributes:
TArray<TCustomAttribute>): Boolean;
begin
  Result := GetAttribute(DISABLE, Attributes) <> Nil;
end;

class function TSerializeParse.IsDisabledRead(const Attributes:
TArray<TCustomAttribute>): Boolean;
begin
  Result := GetAttribute(DISABLEREAD, Attributes) <> Nil;
end;

class function TSerializeParse.IsDisabledWrite(const Attributes:
TArray<TCustomAttribute>): Boolean;
begin
  Result := GetAttribute(DISABLEWRITE, Attributes) <> Nil;
end;

class function TSerializeParse.IsGenerics(Cls: TClass): Boolean;
var
  Info: TGenericsInfo;
  ctx: TRttiContext;
  typ: TRttiType;
begin
  if FGenericsCache.TryGetValue(Cls, Info) then
     Result := Info.IsGeneric
  else
  begin
     Result := False;
     ctx := TRttiContext.Create;
```

```
     try
        typ := ctx.GetType(Cls);
        if typ <> Nil then
           Result := IsGenerics(typ)
     finally
       ctx.Free;
     end;
  end;
end;

class function TSerializeParse.IsGenerics(Cls: TRttiType): Boolean;
var
  C: TClass;
  Mtd: TRttiMethod;
  Info: TGenericsInfo;
  Gt: TGenericsType;
begin
  Result := False;
  C := Cls.AsInstance.MetaclassType;
  if FGenericsCache.TryGetValue(C, Info) then
     Exit(Info.IsGeneric);

  if C.UnitName = GenericsUnit then begin
     Gt := GetGenericType(C);
     if Gt in [gtList, gtObjectList] then begin
        M [блок удален из-за соображений безопасности]
    := RawData

  else if Member is TRttiArrayType then
     TSuperArr(Member).ArrayRawData := RawData;
end;

class procedure TSerializeParse.SetValue<Typ>(var Data: Pointer; Member:
TRttiObject; MIdx: Typ; Val: TValue);
var
  RVal: REVAL;
begin
  if (TypeInfo(Typ) = TypeInfo(Integer) ) then
      case GetMemberTypeInfo(Member, False).Kind of
         tkArray: begin
             Val.ExtractRawData(Data);
             Exit;
         end;

         tkDynArray: begin
             GetValue<String>(GetArrayRawData(Member), Member,
'').SetArrayElement(PInteger(@MIdx)^, Val);
             Exit;
         end;
      end;

  if Member is TRttiProperty  then begin
     RVal := GetREVAL(TRttiProperty(Member).GetAttributes);
     if (RVal <> Nil) and (RVal.CheckEQ(Val)) then
         Val := TValue.FromVariant(RVal.Value);
```

```
      TRttiProperty(Member).SetValue(Data, Val)

  end else if Member is TRttiField then begin
       RVal := GetREVAL(TRttiProperty(Member).GetAttributes);
       if (RVal <> Nil) and (RVal.CheckEQ(Val)) then
           Val := TValue.FromVariant(RVal.Value);
       TRttiField(Member).SetValue(Data, Val);

  end else begin
      if Val.IsObject then
         PPointer(Data)^ := Val.AsObject
      else
         PPointer(Data)^ := Val.GetReferenceToRawData
  end;

end;

class procedure TSerializeParse.WriteCollection(ACollection: TCollection;
IData: ISuperArray);
var
  ItemType: TRttiType;
  Item: TCollectionItem;
  JMembers: IMember;
begin
  with TRttiContext.Create do
      try
        ItemType := GetType(ACollection.ItemClass)
      finally
        Free;
      end;

  if ItemType = Nil then
     raise ESerializeError.CreateFmt('Unknown collection item type
(%s).', [ACollection.ItemClass.ClassName]);

  for JMembers in IData do
      if JMembers.DataType <> dtObject then
         Continue
      else begin
         Item := ACollection.Add;
         WriteMembers(Item, ItemType, JMembers.AsObject);
      end;
end;

class procedure TSerializeParse.WriteGeneric(AObject: TObject; IData:
ISuperArray);
var
  Info: TGenericsInfo;
  Item: TObject;
  JMembers: IMember;
begin
  if IData.DatatYpe = dtNil then
    Exit;

  Info := FGenericsCache[AObject.ClassType];
```

```
  for JMembers in IData do
      if JMembers.DataType <> dtObject then
         Continue
      else
      begin
         Item := ObjectConstructor(Info.Typ.AsInstance.MetaclassType);
         WriteMembers(Item, Info.Typ, JMembers.AsObject);
         Info.AddVal(AObject, Item);
      end;
end;

class procedure TSerializeParse.WriteMember<T, Typ>(Data: Pointer;
Member: Typ;
  RType: PTypeInfo; MemberValue: TRttiObject; IJsonData: IBaseJSON<T,
Typ>);
var
  I: Integer;
  J: NativeInt;
  P: Pointer;
  V: Variant;
  SubVal: TValue;
  SubArr: ISuperArray;
  DataType: TDataType;
  Obj: TObject;
  Ancestor: IJSONAncestor;
begin
  if not IJsonData.Contains(Member) then
     Exit;

  if (RType = TypeInfo(TDateTime)) or (RType = TypeInfo(TDate)) or (RType
= TypeInfo(TTime)) then
  begin
    Ancestor := IJSONData.Ancestor[Member];
    if not (Ancestor.DataType in [dtNull, dtString]) then
       SetValue<Typ>(Data, MemberValue, Member,
TValue.From<TDateTime>(Ancestor.AsVariant))
  end
  else
  case RType.Kind of
    tkInteger:
       SetValue<Typ>(Data, MemberValue, Member,
Integer(IJSonData.I[Member]));

    tkInt64:
       SetValue<Typ>(Data, MemberValue, Member, IJSonData.I[Member]);

    tkChar,  tkWChar:
       if IJsonData.S[Member] > '' then
          SetValue<Typ>(Data, MemberValue, Member,
TValue.From<Char>(IJSonData.S[Member]{$IFDEF
XE2UP}.Chars[CharIndex]{$ELSE}[CharIndex]{$ENDIF}));

    tkString,tkLString, tkWString, tkUString:
       SetValue<Typ>(Data, MemberValue, Member, IJSonData.S[Member]);
```

```
    tkEnumeration:
       if GetMemberTypeInfo(MemberValue) = TypeInfo(Boolean) then
       begin
          SetValue<Typ>(Data, MemberValue, Member, IJSONData.B[Member]);
       end
       else
       begin
          TValue.Make(IJSONData.I[Member],
GetMemberTypeInfo(MemberValue), SubVal );
          SetValue<Typ>(Data, MemberValue, Member, SubVal);
       end;


    tkFloat:
       SetValue<Typ>(Data, MemberValue, Member, IJsonData.F[Member]);

    tkSet:
       WriteSet(Data, MemberValue, IJsonData.A[Member]);

    tkClass:
       begin
          if CheckObject<Typ>(Data, MemberValue, Member, Obj) then
             if (Obj is TStream) then begin
                {$IFDEF SP_STREAM}
                if IJSONData.Null[Member] = jAssigned then
                   WriteStream(TStream(Obj), IJSonData.Ancestor[Member])
                {$ENDIF}
             end else if IsGenerics(Obj.ClassType) then
                WriteGeneric(Obj, IJSONData.A[Member])
             else if IsCollection(Obj.ClassType) then
                WriteCollection(Obj as TCollection, IJSONData.A[Member])
             else
                WriteObject(Obj, IJSonData.O[Member]);
       end;

    tkVariant:
       begin
         V := IJSONData.V[Member];
         if not VarIsNull(V) then
         begin
            TValue.Make(@V, GetMemberTypeInfo(MemberValue), SubVal);
            SetValue<Typ>(Data, MemberValue, Member, SubVal);
         end;
       end;

    tkDynArray, tkArray:
       begin
         if IJSonData.Null[Member] = jAssigned then
         begin
           SetArrayRawData(MemberValue, Data);
           try
             DataType := IJSONData.DataType;
             if DataType = dtArray then begin
                SubArr := IJSONData.AsArray;
                J := IJsonData.AsArray.Length;
```

```
                if RType.Kind = tkDynArray then
                   DynArraySetLength(PPointer(Data)^, RType, 1, @J);
                TValue.Make(Data, Rtype, SubVal);

             end else begin
                J := IJSonData.A[Member].Length;
                SubVal := GetValue<Typ>(Data, MemberValue, Member);
                if RType.Kind = tkDynArray then begin

DynArraySetLength(PPointer(SubVal.GetReferenceToRawData)^,
SubVal.TypeInfo, 1, @J);
                   if (MemberValue is TRttiProperty) and
(PropGetterType(TRttiProperty(MemberValue)) = pgtMethod) then begin
                      WriteMember<IJSONArray,
Integer>(SubVal.GetReferenceToRawData,
                                              0,
                                              RType,

TRttiProperty(MemberValue).PropertyType,
                                              IJSONData.A[Member]);
                      SetValue<String>(Data, MemberValue,'', SubVal );
                      Exit;

                   end else
                      SetValue<String>(Data, MemberValue,'', SubVal );

                end;
             end;

             for I := 0 to J-1 do begin
                 if DataType <> dtArray then
                    SubArr := IJSONData.A[Member];
                 WriteMember<IJSONArray, Integer>
                            (SubVal.GetReferenceToRawArrayElement(I),
                             I,
                             GetMemberTypeInfo(MemberValue),
                             MemberValue,
                             SubArr);
             end;

             if DataType = dtArray then
                SubVal.ExtractRawData(Data)
             else SetValue<String>(Data, MemberValue,'', SubVal );

           finally
             ClearArrayRawData(MemberValue);
           end;
         end;
       end;

    tkRecord:
       begin
         if (MemberValue.ClassType = TRttiDynamicArrayType) or
(MemberValue.ClassType = TRttiArrayType) then
```

```
            WriteRecord(GetMemberTypeInfo(MemberValue), Data,
IJSonData.O[Member])
         else begin
            P := IValueData(TValueData( GetValue<Typ>(Data, MemberValue,
Member) ).FValueData).GetReferenceToRawData;
            WriteRecord(GetMemberTypeInfo(MemberValue), P,
IJSonData.O[Member]);
            TValue.Make(P, GetMemberTypeInfo(MemberValue), SubVal);
            SetValue<Typ>(Data, MemberValue, Member, SubVal );
         end;
       end;

    tkInterface:
       if (TypeInfo(ISuperObject) = GetMemberTypeInfo(MemberValue)) And
(IJsonData.Ancestor[Member].DataType = dtObject) then
          SetValue<Typ>(Data, MemberValue, Member,
TValue.From<ISuperObject>(IJsonData.O[Member].Clone))
       else
       if (TypeInfo(ISuperArray) = GetMemberTypeInfo(MemberValue)) And
(IJsonData.Ancestor[Member].DataType = dtArray) then
          SetValue<Typ>(Data, MemberValue, Member,
TValue.From<ISuperArray>(IJsonData.A[Member].Clone));
  end;
end;

class procedure TSerializeParse.GetAliasName(const Attributes:
TArray<TCustomAttribute>; var Result: String);
var
  Attr: Alias;
begin
  Attr := Alias(GetAttribute(Alias, Attributes));
  if Assigned(Attr) then
     Result := Attr.Name;
end;

class procedure TSerializeParse.WriteMembers(Data: Pointer; aType:
TRttiType;
  IJsonData: ISuperObject);
var
  Prop: TRttiProperty;
  Field: TRttiField;
  MemberName: String;
begin
  for Prop in aType.GetProperties do
      if Prop.PropertyType <> Nil then
      begin
         if not (Prop.Visibility in TSerializeParseOptions.Visibilities)
then Continue;
         MemberName := Prop.Name;
         if IsDisabled(Prop.GetAttributes) or
IsDisabledRead(Prop.GetAttributes) then
            Continue;
         GetAliasName(Prop.GetAttributes, MemberName);
         WriteMember<IJSONObject, String>(Data, MemberName,
Prop.PropertyType.Handle, TSuperProperty(Prop), IJSonData);
```

```
      end;
  for Field in aType.GetFields do
      if Field.FieldType <> Nil then
      begin
         if not (Field.Visibility in TSerializeParseOptions.Visibilities)
then Continue;
         MemberName := Field.Name;
         if IsDisabled(Field.GetAttributes) or
IsDisabledRead(Field.GetAttributes) then
            Continue;
         GetAliasName(Field.GetAttributes, MemberName);
         WriteMember<IJSONObject, String>(Data, MemberName,
Field.FieldType.Handle, TSuperField(Field), IJSonData);
      end;
end;

class procedure TSerializeParse.WriteObject(AObject: TObject;
  IData: ISuperObject);
var
  Ctx: TRttiContext;
  Typ: TRttiType;
begin
  Ctx := TRttiContext.Create;
  try
    Typ := Ctx.GetType(AObject.ClassType);
    if (not Assigned(Typ)) or (IData.DataType = dtNil) then Exit;
    WriteMembers(AObject, Typ, IData);
  finally
    Ctx.Free;
  end;
end;

class procedure TSerializeParse.WriteRecord(Info: PTypeInfo; ARecord:
Pointer;
  IData: ISuperObject);
var
  Ctx: TRttiContext;
  Typ: TRttiType;
begin
  Ctx := TRttiContext.Create;
  try
    Typ := Ctx.GetType(Info);
    if (not Assigned(Typ)) or (IData.DataType = dtNil) then
      Exit;
    WriteMembers(ARecord, Typ, IData);
  finally
    Ctx.Free;
  end;
end;

class procedure TSerializeParse.WriteRecordEx<T>(Rec: T;
  IData: ISuperObject);
begin
   with TValue.From<T>(Rec) do
        WriteRecord(TypeInfo, GetReferenceToRawData, IData);
```

```
end;

class procedure TSerializeParse.WriteSet(Data: Pointer; Member:
TRttiObject;
  IJSONData: ISuperArray);
var
  Sets: TIntegerSet;
  I: Integer;
  Val: TValue;
begin
  Sets := [];
  for I := 0 to IJSONData.Length -1 do
      Include(Sets, IJSONData.I[I]);
  TValue.Make(Integer(Sets), GetMemberTypeInfo(Member), Val);
  SetValue<String>(Data, Member, '', Val);
end;

{$IFDEF SP_STREAM}
function Base64Decode(const EncodedText: string): TBytesStream;
var
  Decoder: TIdDecoderMIME;
begin
  Decoder := TIdDecoderMIME.Create(nil);
  try
    Result := TBytesStream.Create;
    try
      Decoder.DecodeBegin(Result);
      Decoder.Decode(EncodedText);
      Decoder.DecodeEnd;
    except
      Result.Free;
      raise;
    end;
  finally
    Decoder.Free;
  end;
end;

class procedure TSerializeParse.WriteStream(AStream: TStream; IData:
IJSONAncestor);
var
  Setter: TBytesStream;
begin
  Setter := Base64Decode((IData as TJSONString).Value);
  try
    Setter.Position := 0;
    AStream.CopyFrom(Setter, Setter.Size);
  finally
    Setter.Free;
  end;
end;
{$ENDIF}

{ TSuperRecord<T> }
```

```
class function TBaseSuperRecord<T>.AsJSON(Rec: T): String;
begin
  Result := AsJSONObject(Rec).AsJSON;
end;

class function TBaseSuperRecord<T>.FromJSON(JSON: String): T;
begin
   Result := FromJSON(SO(JSON));
end;

class function TBaseSuperRecord<T>.AsJSONObject(Rec: T): ISuperObject;
begin
  Result := XSuperObject.TSerializeParse.ReadRecordEx<T>(Rec);
end;

class function TBaseSuperRecord<T>.FromJSON(JSON: ISuperObject): T;
var
  Val: TValue;
  P: Pointer;
begin
  FillChar(Result, SizeOf(T), 0);
  Val := TValue.From<T>(Result);
  P := IValueData(TValueData(Val).FValueData).GetReferenceToRawData;
  TSerializeParse.WriteRecord(Val.TypeInfo, P, JSON);
  Result := T(P^);
end;

{ TJSONValueHelper }

function TJSONValueHelper.ValueEx<T>: Variant;
var
  Valuable: Boolean;
  pV: PTypeInfo;
const
  Int = 0;
  Str = '';
begin
  Valuable := (Self <> Nil) and not isNull;
  pV := TypeInfo(T);
  if pV = TypeInfo(Int64) then begin
     if Valuable then
        Result := (Self as TJSONInteger).Value
     else
        Result := Int;
  end
  else
  if pV = TypeInfo(Double) then begin
     if Valuable then
        Result := (Self as TJSONFloat).Value
     else
        Result := Int
  end
  else
  if pV = TypeInfo(Boolean) then begin
     if Valuable then
```

```
        Result := (Self as TJSONBoolean).Value
      else
        Result := False
  end
  else
  if pV = TypeInfo(String) then
      if Valuable then
        Result := (Self as TJSONString).Value
      else
        Result := Str
end;

{ TSuperExpression }

constructor TSuperExpression.Create(Base: IJSONAncestor; const Expr:
String; const BlockException: Boolean);
begin
  FInterpreter := TJSONInterpreter.Create(Expr, Base, BlockException);
  inherited Create(FInterpreter.ReadExpression);
end;

destructor TSuperExpression.Destroy;
begin
  FInterpreter.Free;
  inherited;
end;

{ TCast }

constructor TCast.Create(Base: IJSONAncestor);
begin
  FJSON := Base;
  FName := '';
end;

constructor TCast.Create(Base: IJSONPair);
begin
  FJSON := Base.JSONValue;
  FName := Base.Name;
end;

class function TCast.CreateFrom<T>(Base: T): ICast;
var
  IFace: IInterface;
begin
  IFace := TValue.From<T>(Base).AsInterface;
  if IFace is TJSONAncestor then
     Result := TCast.Create(IFace as TJSONAncestor)
  else
  if IFace is TJSONPair then
     Result := TCast.Create(IFace as TJSONPair)
  else
     Result := TCast.Create(TJSONAncestor(Nil));
end;
```

```
destructor TCast.Destroy;
begin
  FJSON := Nil;
  inherited;
end;

function TCast.GetArray: ISuperArray;
begin
  if not Assigned(FJSON) then
     Result := Nil
  else
     Result := TSuperArray.Create(FJSON as TJSONArray);
end;

function TCast.GetBoolean: Boolean;
begin
  if not Assigned(FJSON) then
     Result := False
  else
     Result := TJSONBoolean(FJSON).Value;
end;

function TCast.GetDataType: TDataType;
begin
  if FJSON = Nil then
     Result := dtNil
  else if FJSON is TJSONNull then
     Result := dtNull
  else if FJSON is TJSONString then
     Result := dtString
  else if FJSON is TJSONInteger then
     Result := dtInteger
  else if FJSON is TJSONFloat then
     Result := dtFloat
  else if FJSON is TJSONBoolean then
     Result := dtBoolean
  else if FJSON is TJSONObject then
     Result := dtObject
  else if FJSON is TJSONArray then
     Result := dtArray
  else if FJSON is TJSONDateTime then
     Result := dtDateTime
  else if FJSON is TJSONDate then
     Result := dtDate
  else if FJSON is TJSONTime then
     Result := dtTime
  else
     raise SOException.Create('Unknown JSON Type');
end;

function TCast.GetDate: TDate;
begin
  if not Assigned(FJSON) then
     Result := 0
  else
```

```
     Result := TJSONDate(FJSON).Value;
end;

function TCast.GetDateTime: TDateTime;
begin
  if not Assigned(FJSON) then
     Result := 0
  else
     Result := TJSONDateTime(FJSON).Value;
end;

function TCast.GetFloat: Double;
begin
  if not Assigned(FJSON) then
     Result := 0
  else
     if FJSON is TJSONInteger then
        Result := TJSONInteger(FJSON).Value
     else
        Result := TJSONFloat(FJSON).Value;
end;

function TCast.GetInteger: Int64;
begin
  if not Assigned(FJSON) then
     Result := 0
  else
  if DataType <> dtInteger then
     Result := StrToIntDef(VarToStr(GetVariant), 0)
  else
     Result := TJSONInteger(FJSON).Value;
end;

function TCast.GetName: String;
begin
  Result := FName;
end;

function TCast.GetObject: ISuperObject;
begin
  if not Assigned(FJSON) then
     Result := Nil
  else
     Result := TSuperObject.Create(FJSON as TJSONObject);
end;

function TCast.GetString: String;
begin
  if not Assigned(FJSON) then
     Result := ''
  else
     if FJSON is TJSONString then
        Result := TJSONString(FJSON).Value
     else
        Result := VarToStr(FJSON.AsVariant);
```

```
end;

function TCast.GetTime: TTime;
begin
  if not Assigned(FJSON) then
     Result := 0
  else
     Result := TJSONTime(FJSON).Value;
end;

[блок удален из-за соображений безопасности]

{ ReNameField }

constructor Alias.Create(const AName: String);
begin
  FName := AName;
end;

{ REVAL }

function REVAL.CheckEQ(Val: TValue): Boolean;
begin
  case FOption of
    roNone:
      Result := Val.AsVariant = FEqual;
    roEmptyArrayToNull:
      Result := Val.GetArrayLength = 0;
    else raise Exception.CreateFmt('Unknown option: %d', [Ord(FOption)]);
  end;
end;

constructor REVAL.Create(EQVal, NewVal: Integer);
begin
  FOption := roNone;
  FEqual := EQVal;
  FValue := NewVal;
end;

constructor REVAL.Create(EQVal, NewVal: String);
begin
  FOption := roNone;
  FEqual := EQVal;
  FValue := NewVal;
end;

constructor REVAL.Create(EQVal, NewVal: Double);
begin
  FOption := roNone;
  FEqual := EQVal;
  FValue := NewVal;
end;

constructor REVAL.Create(EQVal: String);
begin
```

```
  FOption := roNone;
  FEqual := EQVal;
  FValue := Variants.Null;
end;

constructor REVAL.Create(EQVal, NewVal: Boolean);
begin
  FOption := roNone;
  FEqual := EQVal;
  FValue := NewVal;
end;

constructor REVAL.Create(EQVal: Integer);
begin
  FOption := roNone;
  FEqual := EQVal;
  FValue := Variants.Null;
end;

constructor REVAL.Create(EQVal: Double);
begin
  FOption := roNone;
  FEqual := EQVal;
  FValue := Variants.Null;
end;
[блок удален из-за соображений безопасности]
        end;
  except
    Ctx.Free;
    raise;
  end;
end;


class function TJSON.Parse<T>(JSON: ISuperArray): T;
var
  Ctx: TRttiContext;
  _PResult: Pointer;
  Typ: TRttiType;
begin
  Ctx := TRttiContext.Create;
  try
    Typ := Ctx.GetType(TypeInfo(T));
    if not Assigned(Typ) then Exit(Default(T));
    _PResult := @Result;
    TSerializeParse.WriteMember<IJSONArray, Integer>(
                         _PResult,
                         0,
                         Typ.Handle,
                         Typ,
                         JSON);
   finally
     Ctx.Free;
   end;
end;
```

```
class function TJSON.Stringify(Value: TValue; Indent, UniversalTime:
Boolean): String;
begin
  Result := SuperObject(Value).AsJSON(Indent, UniversalTime);
end;

{$IFDEF SP_DATASET}

class function TJSON.Stringify(Value: TDataSet): String;
begin
  Result := SuperObject(Value).AsJSON(False, True);
end;

class function TJSON.SuperObject(Value: TDataSet): ISuperObject;
var
  I, J, Z: Integer;
  Rec: ISuperObject;
  Return: ISuperArray;
  Bookmark: TBookmark;
  ABytes: TArray<Byte>;
begin
  Return := SA;
  Bookmark := Value.Bookmark;
  Value.DisableControls;
  try
    if not Value.Active then
       Value.Active := True;
    Value.First;
    for I := 0 to Value.RecordCount - 1 do begin
        Rec := SO;
        for J := 0 to Value.FieldCount - 1 do with Value.Fields.Fields[J]
do begin
            case DataType of
              ftString:
                 Rec.S[FieldName] := AsString;
              ftSmallint, ftInteger:
                 Rec.I[FieldName] := AsInteger;
              ftBoolean:
                 Rec.B[FieldName] := AsBoolean;
              ftFloat:
                 Rec.F[FieldName] := AsFloat;
              ftCurrency:
                 Rec.F[FieldName] := AsCurrency;
              ftDate:
                 Rec.Date[FieldName] := TDate(AsDateTime);
              ftTime:
                 Rec.Time[FieldName] := TTime(AsDateTime);
              ftDateTime:
                 Rec.D[FieldName] := AsDateTime;
              ftWideString:
                 Rec.S[FieldName] := AsWideString;
              ftLargeint, ftAutoInc:
                 Rec.I[FieldName] := AsLargeInt;
              ftVariant:
```

```
                Rec.V[FieldName] := AsVariant;
              ftShortint:
                Rec.I[FieldName] := AsInteger;
              ftByte: begin
                ABytes := AsBytes;
                with Rec.A[FieldName] do
                     for Z := 0 to High(ABytes) do
                         Add(ABytes[Z]);
              end;
              ftExtended:
                Rec.D[FieldName] := AsExtended;
            end;
        end;
        Return.Add(Rec);
        Value.Next;
    end;
  finally
    Value.Bookmark := Bookmark;
    Value.EnableControls;
  end;
  Result := TSuperObject.CreateCasted(Return.Self);
end;
{$ENDIF}

class function TJSON.Stringify<T>(Value: T; Indent: Boolean;
UniversalTime: Boolean): String;
begin
  Result := SuperObject<T>(Value).AsJSON(Indent, UniversalTime);
end;

class function TJSON.Parse<T>(const Value: String): T;
begin
  Result := Parse<T>(SO(Value));
end;

class function TJSON.SuperObject<T>(Value: T): ISuperObject;
begin
  Result := SuperObject(TValue.From<T>(Value));
end;

initialization

  GenericsUnit := TEnumerable<Boolean>.UnitName;

end.
```