**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ ДОНЕЦКОЙ НАРОДНОЙ РЕСПУБЛИКИ**

**ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

**«ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**ФАКУЛЬТЕТ Компьютерных наук и технологий**

**КАФЕДРА Автоматизированных систем управления**

**НАПРАВЛЕНИЕ «Информационные системы и технологии»**

Допустить к защите

Заведующий кафедрой АСУ

\_\_\_\_\_\_\_\_\_\_\_ Привалов М.В.

(Подпись) (ФИО)

«\_\_\_» \_\_\_\_\_\_\_\_\_ 2016 г.

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к бакалаврской работе**

**НА ТЕМУ: РАЗРАБОТАТЬ КОМПЬЮТЕРИЗИРОВАННУЮ ПОДСИСТЕМУ НАЧИСЛЕНИЯ И УЧЕТА ПЛАТЕЖЕЙ ЗЕМЕЛЬНОГО НАЛОГА В УСЛОВИЯХ РАЙОННОЙ НАЛОГОВОЙ ИНСПЕКЦИИ** *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**АВТОР БАКАЛАВРСКОЙ РАБОТЫ**

Казанцев Максим алеклеевич \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Фамилия, имя, отчество) (Подпись)

**РУКОВОДИТЕЛЬ БАКАЛАВРСКОЙ РАБОТЫ**

ст.преп. андриевская н.к.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Должность, ученая степень, ученое звание, фамилия, имя, отчество) (Подпись)

**КОНСУЛЬТАНТЫ:**

|  |
| --- |
| ОХРАНА ТРУДА, К.Х.Н., ДОЦЕНТ, БУТУЗОВ Г.Н. |
| (Наименование раздела (подраздела), должность, ученая степень, звание, ФИО, подпись) |
|  |
|  |

Донецк - 2016

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ ДОНЕЦКОЙ НАРОДНОЙ РЕСПУБЛИКИ**

**ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

**«ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**ФАКУЛЬТЕТ Компьютерных наук и технологий**

**КАФЕДРА Автоматизированных систем управления**

**НАПРАВЛЕНИЕ «Информационные системы и технологии»**

УТВЕРЖДАЮ:

Заведующий кафедрой АСУ

\_\_\_\_\_\_\_\_\_\_\_ Привалов М.В.

(Подпись) (ФИО)

«\_\_\_» \_\_\_\_\_\_\_\_\_ 2016 г.

**ЗАДАНИЕ**

на бакалаврскую работу

СтудентуКАЗАНЦЕВУ МАКСИМУ АЛЕКСЕЕВИЧУ**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Фамилия, имя, отчество)

1.Тема бакалаврской работы: Разработать компьютеризированную подсистему начисления

и учета платежей земельного налога в условиях районной налоговой инспекции

( в случае выполнения комплексного проекта (работы) в скобках указывается «комплексный (ая)»)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

утверждена приказом от «\_\_25\_\_\_\_» \_\_\_04\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2016\_\_\_г. №\_\_428-14\_\_\_\_\_

2. Исходные данные к бакалаврской работе\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**\_\_\_\_\_\_\_\_\_материалы курсовых проектов и практик, в том числе преддипломной\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

3. Содержание пояснительной записки к бакалаврской работе \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1) Анализ объекта управления

2) Определение функциональной структуры проектируемой подсистемы

3) Разработка информационного обеспечения

4) Рекомендации по разработке математического и алгоритмического обеспечения

5) Программное обеспечение

6) Охрана труда

4. Перечень графического материала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

5. Данные о консультантах, с указанием разделов пояснительной записки

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Раздел | Консультант | Задание выдал | | Задание принял | |
| подпись | дата | подпись | дата |
| 1 | ст. преп. Андриевская Н.К. |  |  |  |  |
| 2 | ст. преп. Андриевская Н.К. |  |  |  |  |
| 3 | ст. преп. Андриевская Н.К. |  |  |  |  |
| 4 | ст. преп. Андриевская Н.К. |  |  |  |  |
| 5 | ст. преп. Андриевская Н.К. |  |  |  |  |
| 6 | к.х.н., доцент, Бутузов Г.Н. |  |  |  |  |

6. Срок сдачи студентом законченной бакалаврской работы\_\_\_\_\_31.05.2016\_\_\_\_\_\_\_\_\_\_\_\_\_

7. Дата выдачи задания на бакалаврскую работу \_\_\_\_\_\_\_\_\_18.04.2016\_г.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель ст. преп. Андриевская Н.К. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Фамилия, имя, отчество) (Подпись)

Задание принял к исполнению \_\_\_\_\_\_\_18.04.2016 г.\_\_\_\_\_\_\_ . \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Дата) (Подпись)

КАЛЕНДАРНЫЙ ПЛАН

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Название этапов бакалаврской работы | Срок выполнения этапов работы | Отметка о выполнении |
| 1 | Анализ объекта управления |  |  |
| 2 | Определение и разработка функциональной структуры проектируемой подсистемы | 1.05.2015 | выполнено |
| 3 | Разработка информационных моделей | заполнить | заполнить |
| 4 | Разработка математического и алгоритмического обеспечения | заполнить | заполнить |
| 5 | Разработка программного обеспечения | заполнить | заполнить |
| 6 | Охрана труда | заполнить | заполнить |
| 7 | Оформление пояснительной записки | заполнить | заполнить |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

Руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| РЕФЕРАТ  Пояснительная записка к бакалаврской работе содержит: 138 с., 34 рис.,  22 табл., 9 приложений, 6 источников.  Целью выпускной квалификационной работы является разработка компьютеризированной подсистемы начисления и учета платежей земельного налога в условиях районной налоговой инспекции.  Функционал системы определяется требованиями, которые были сформулированы на этапе анализа предметной области и аналогичных программных продуктов.  В результате проектирования была разработана БД и ПО, предоставляющие налоговому инспектору возможность комфортно и продуктивно выполнять ввод и обработку данных, а также формировать выходные документы.  Про алгоритмтическое и охрану труда забыл!! Добавь предложение..  ПОДСИСТЕМА, ДЕКЛАРАЦИЯ, УЧЕТ, ФУНКЦИЯ, МОДЕЛЬ, ДАННЫЕ, ПРОГРАММА, ФОРМА, ДОКУМЕНТ | | | | | | | | | | |
|  |  |  |  |  | 09.03.01.Б2016.12\113982.00.00 ПЗ | | | | | |
|  |  |  |  |  |
| Изм. | Лист | № документа | Подпись | Дата |
| Студент | | Казанцев М.А. |  |  | Разработать компьютеризированную подсистему начисления  и учета платежей земельного налога в условиях районной налоговой инспекции  Пояснительная записка | Лит | | | Лист | Листов |
| Рук. | | Андриевская Н.К. |  |  | У |  |  | 1 | 137 |
| Консульт. | |  |  |  | ДонНТУ, ФКНТ,  Кафедра АСУ, гр.ИУС-12а | | | | |
| Консульт. | |  |  |  |
| Н. контр. | | Привалов М.В. |  |  |

**Перечень замечаний нормоконтролёра к бакалаврской работе**

Студента Казанцева Максима Алексеевича, группы ИУС-12а

|  |  |  |  |
| --- | --- | --- | --- |
| Обозначение документа | Документ | Условное обозначение | Содержание замечания |
|  |  |  |  |

Дата \_\_\_\_\_\_\_\_\_\_\_\_ Подпись \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Ф.И.О.)

ABSTRACT

The main purpose of the final qualifying work is to develop an automatized calculating and accounting subsystem for land tax within the local tax inspection.

The subsystem contains database and concomitant software, which gives an opportunity for tax inspector to interact with information arrays.

Functionality of the system was identified by the requirements, which have been formulated during the analysis of the subject area and similar software products.

The current stage of implementation affects key aspects of using the system based on the workstation. However, data security and networking are ignored for now.

**СОДЕРЖАНИЕ**

**ВВЕДЕНИЕ**

Проектом является система, обеспечивающая учет уплаты налога на землю налогоплательщиками, которыми являются юридические лица. Система эксплуатируется работниками районной налоговой инспекции, чья деятельность связана с налогообложением земельных участков.

Данная предметная область далеко не нова и неоднократно подвергалась компьютеризации некоторых своих аспектов различно степени и глубины. Об этом свидетельствует внушительное количество программных пакетов и разнообразных систем учетного типа и не только. Некоторые такие системы являются успешными коммерческими проектами, которые находятся в эксплуатации уже не один год. Стоит отметить, что функционирование подобных систем тесно связано с законодательством административной единицы территории того государства, где система находится в использовании. Как правило, рынок таких продуктов монополизируется наиболее влиятельным поставщиком, в следствии чего, цена на системы становится достаточно высокой. Такое положение дел затрудняет покупку и использование подобных программ представителями малого бизнеса и небольших предприятий.

Актуальности данной тематике также предает тот факт, что текущая ситуация в правовом поле региона достаточно туманна. Фактически, это предоставляет некоторую свободу действий в реализации функционала подсистемы.

Целью проекта является разработка подсистемы с использованием архитектуры и технологий, которые минимизируют ее стоимость и сделают максимально простой в использовании, тем самым фактически исключив необходимость поиска и найма высококвалифицированных кадров.

При разработке подсистемы за основу взяты правовое поле и механизмы налогообложения земельных участков соседнего государства.

В разделе «Анализ объекта управления» приведена характеристика объекта компьютеризации, проведен детальный анализ аналогичных подсистем, описана работа наиболее примечательных из них, выявлены проблемы их использования, которые актуализируют данный проект по разработке аналога этих систем, лишенного присущих им минусов.

В разделе «Определение функциональной структуры проектируемой компьютеризированной подсистемы» обоснован выбор CASE – средства схематического представления разрабатываемой системы, также были описаны функции, которые подлежат компьютеризации, при их описании широко использовались схематические представления, оформленные с использованием выбранного CASE – средства.

В разделе «Разработка информационного обеспечения» приведен перечень СУБД, составлено детальное сравнение двух из них, выбор которых наиболее предпочтителен. В разделе описаны системы классификации и кодирования всех кодовых обозначений, которые присутствуют на данном этапе разработки подсистемы.

На основе анализа предметной области были выделены сущности, определены связи между ними и на основе этого составлены модели данных.

Спроектированная таким образом база была полностью реализована в среде выбранной СУБД. Также была рассмотрена организация сбора и обработки информации, описание которой сопровождалось представлением специальной схемы.

Раздел «Рекомендации по разработке математического и алгоритмического обеспечения» содержит более детальное описание основных функций системы, в процессе которого были описаны входные и выходные документы, были составлены алгоритмы выполнения этих функций, а также сформулированы требования к формату данных, которые используются в ходе работы системы.

В разделе «Программное обеспечение» достаточно детально описываются разрабатываемые программные модули, пользовательский интерфейс, а также технологии, которые обеспечили реализацию связи с базой данных и формирование отчетов на основе информации из нее. В процессе описания были выделены абстракции, проанализировано их поведение. На основе проведенного анализа составлены диаграммы абстракций и классов.

В конце раздела детально описывается результат тестирования программы на контрольном примере.

Раздел «Охрана труда» содержит свод правил, выполнение которых крайне важно для обеспечения эффективной и безопасной работы сотрудников районной налоговой инспекции.

**ПЕРЕЧЕНЬ СОКРАЩЕНИЙ, УСЛОВНЫХ ОБОЗНАЧЕНИЙ, СИМВОЛОВ, ЕДИНИЦ И ТЕРМИНОВ**

НДО – нормативно – денежная оценка

ЕГРПОУ – Единый Государственный Реестр Предприятий и Организаций Украины

КОАУТУУ – классификатор объектов административно – территориального устройства Украины

КВЭД – код вида экономической деятельности

СУБД – система управления базами данных

БД – база данных

ИС – информационная система

**1 АНАЛИЗ ОБЪЕКТА УПРАВЛЕНИЯ**

* 1. **Характеристика объекта компьютеризации**

Анализируемый объект компьютеризации тесно связан с работой районной налоговой инспекции, точнее с ее функционалом, ориентированным на учет налогов на земельные участки, собственниками которых являются налогоплательщики, в частности физические и юридические лица.

Функционал автоматизируемой системы подразумевает расчет налога на земельные участки, находящиеся в использовании у налогоплательщика. Согласно данным из правоустанавливающих документов на землю (см. Приложение Б), налоговой декларации (см. Приложения А), а также справки о нормативно – денежной оценке земельного участка происходит расчет недоимки, штрафов, пени, а также самого налога с учетом предоставляемых налогоплательщику льгот.

Использование системы осуществляет налоговый инспектор для юридических лиц, действия которого регламентируется действующим законодательством:

* Земельным Кодексом Украины;
* Налоговым Кодексом Украины;
* [Законом Украины от 03.07.92 г. № 2535-XII "О плате за землю"](file:///C:\Users\hp\AppData\Local\Temp\Avers\ZK_VRU_2535-XII_92R.HTM);
* [Законом Украины от 27.11.2003 г. № 1344-IV "О Государственном бюджете Украины на 2004 год"](file:///C:\Users\hp\AppData\Local\Temp\Avers\ZK_VRU_1344-IV_03U.HTM);
* [Порядком представления налогового расчета земельного налога, утвержденного приказом ГНАУ от 26.10.2001 г. № 434](file:///C:\Users\hp\AppData\Local\Temp\Avers\PR_GNA_434_01U.HTM);

Выполняемые системой функции:

* Регистрация и учет налогоплательщиков. Входящие документы анализируются и данные из них заносятся в электронный реестр.
* Учет налогообложения. Согласно данным из документации, которая подавалась в налоговую инспекцию юридическим лицом, а также информации из реестра налогоплательщиков, происходит расчет суммы налога с учетом льгот (данные о льготах и налоговых ставках берутся из соответствующих справочников). Результаты расчетов заносятся в налоговую квитанцию. Налоговая декларация, после анализа данных и проверки корректности заполнения импортируется в базу данных.
* Учет погашения просроченных налоговых обязательств. В случае если у налогоплательщика имеются задолженности – происходит формирование расчетов недоимки, пени и штрафов. Результаты расчетов заносятся в уточненную декларацию по налоговым обязательствам. Формируются отсрочка по выплате и календарный график погашения недоимки.
* Формирование итоговых отчетов в органы статистики и вышестоящие инстанции.
  1. **Описание работы существующей ИС**

Одним из основных средств работы налогового инспектора является пакет «M.E.Doc».

Функционал пакета предусматривает ввод и отправку налогоплательщиком, а, в последствии, и сбор информации об объектах налогообложения налоговой инспекцией, проверку корректности введенных данных, а также проверку на наличие просроченных налоговых обязательств.

При получении документов от налогоплательщика, отдел информатизации процессов формирует из них электронный пакет документации, который будет направлен в другие отделы.

Основными документами, которые обрабатываются в процессе работы с «M.E.Doc» являются:

* Налоговая декларация – содержит данные об объекте налогообложения: код по ЕГРПОУ, код ВЭД, информацию о налогоплательщике и т.д.;
* Правоустанавливающие документы на землю – являются юридическим подтверждением того, что налогоплательщик обладает правом на владение объектом налогообложения;
* Справка о нормативно – денежной оценке земельного участка – содержит информацию о фактической стоимости единицы площади, а также земельном участке в целом в денежном эквиваленте;

Схема документооборота приведена на рисунке 1.1. Формы вышеупомянутых документов приведены в приложениях А – В.

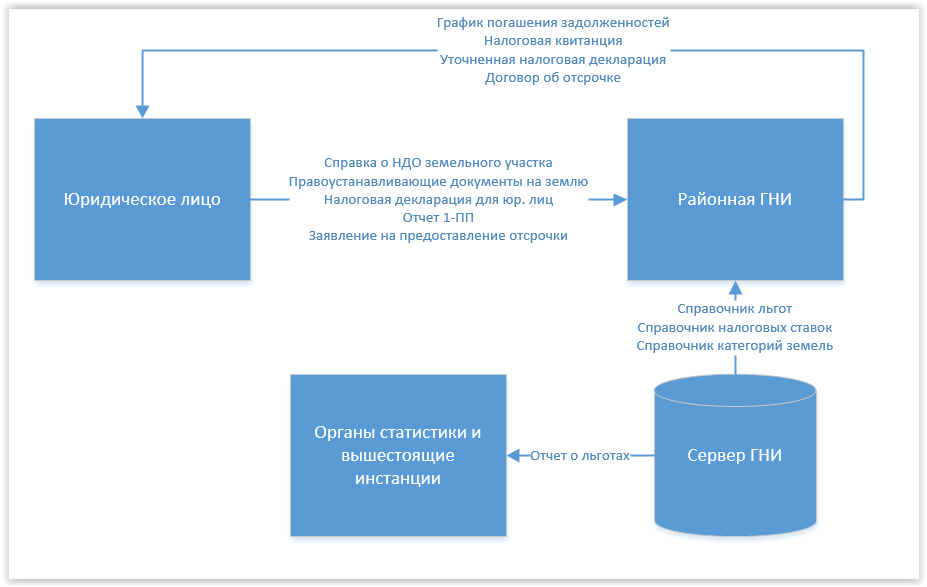
****

Рисунок 1.1 – Схема документооборота

Пакет программного обеспечения «M.E.Doc» обладает достаточным уровнем автоматизации работы, он способен проверять корректность полученных данных, рассчитывать налог, а также определять величину просроченных налоговых обязательств (при их наличии у налогоплательщика).

«M.E.Doc» работает на базе операционных систем Windows XP, Windows Server 2003 или же их более поздних версий. Для стабильного функционирования необходим процессор с минимальной тактовой частотой 1,8-2 ГГц, не менее 1,5 ГБ места на жестком диске и 2 ГБ оперативной памяти (от 30 МБ на пользователя, в случае с сервером), для связи требуется сеть со скоростью от 10 МБит/с. Также, для полноценного взаимодействия с вышеупомянутым программным обеспечением, необходим цветной графический дисплей (разрешение 1024 х 768), лазерный или струйный принтер и клавиатура с мышью.

Для комфортной работы с программой на компьютере должен быть установлен пакет MS Office (программы Word и Excel), а в региональных настройках операционной системы – русский или украинский язык.

К недостаткам «M.E.Doc» можно отнести невозможность расчета пени и арендной платы за земельный участок. Наиболее вероятно, что это вызвано сбоями в расчетах внутри программы или же отсутствием данного функционала, что, с учетом наличия средств для его реализации, весьма странно. Такие проблемы приводят к увеличению затрачиваемого на расчеты времени и, как следствие, к снижению общей производительности персонала предприятия и налоговой инспекции.

* 1. **Анализ аналогичных подсистем**

К рассматриваемому объекту компьютеризации применимы следующие разработанные автоматизированные системы:

* «M.E.Doc»
* «[Арт](https://www.google.com.ua/url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&cad=rja&uact=8&ved=0CCsQFjACahUKEwi2wq7BuPbHAhWF6CwKHQz8De0&url=http%3A%2F%2Fart-zvit.com.ua%2F&usg=AFQjCNGpNGTbvXym_q3OFdi4fQ5XujO7Hw) – Звiт +»
* «Taxer»
* «1С – Звiт»
* «Соната»
* «iFin»

Все вышеперечисленные пакеты программного обеспечения являются коммерческими и требуют оплаты перед использованием, однако у «M.E.Doc» есть особенность, которая выгодно его отличает от остальных. Это коммерческая программа с демо – версией, которая полностью функциональна и обладает лишь некоторыми ограничениями, связанными с экспортом, распечатыванием и электронной цифровой подписью документов.

Компьютерная программа «M.E.Doc» (My Electronic Document) разработана командой компании «Интеллект – сервис». Это программный продукт для предприятий различных сфер деятельности, независимо от системы налогообложения.

«M.E.Doc» – не просто программа для формирования и сдачи отчетности, это полноценная система электронного документооборота, с помощью которой можно работать с документами различных типов и назначений: отчетами, налоговыми накладными, договорами, счетами, актами и др.

При помощи данного программного пакета можно:

* создать все необходимые документы;
* подписать документы при помощи ЭЦП — электронно-цифровой подписи;
* обменяться с контрагентами налоговыми накладными и квитанциями об их регистрации;
* запросить и получить Выписки из реестра налоговых накладных;
* проверить, зашифровать и сохранить электронные документы в единой системе;
* за считанные секунды отправить отчеты в электронном виде в:  
  — Государственную фискальную службу (налоговая инспекция);  
  — Пенсионный фонд Украины;  
  — Государственный комитет статистики;
* получить квитанции о приеме и обработке отправленных документов (отчетов, налоговых накладных, и др.).

«M.E.Doc» имеет модульную структуру, благодаря чему имеется возможность выбрать необходимый набор функций при покупке:

* Модуль «Электронный документооборот» – регистрация налоговых накладных в ЕРНН и обмен первичными документами с контрагентами.
* Модуль «Электронный документооборот с дополнительными возможностями» - администрирование регистрации горючего, создание сводных налоговых накладных и выданных на основе полученных, обмен документами с вложениями (договора и пр.)
* Модуль «Отчетность» – подача всех видов отчетности во все контролирующие органы.
* Модуль «Корпорация» – консолидация отчетности предприятий с разветвленной структурой.
* Модуль «Зарплата» – расчет и начисление заработной платы сотрудников, учет и управление персоналом.
* Модуль «Банковские счета» – автоматизация уведомлений об открытии или закрытии счетов.
* «СОТА» – веб-сервис для подачи отчетности юридическими и физическими лицами-предпринимателями на единой системе налогообложения.

Все документы, созданные в программе «M.E.Doc» и подписанные электронной цифровой подписью, являются оригиналами документов (Законы Украины «Об электронной цифровой подписи» № 852-IV от 22.05.2003г. и «Об электронных документах и электронном документообороте» № 851-IV от 22.05.2003г.).

При разработке программного пакета использовалась архитектура типа «клиент – сервер».

«M.E.Doc» использует Oracle версии 10.2 или более поздней в качестве СУБД.

Для нормальный работы программы, а также установки и обновления, необходимо наличие следующих установленных компонентов:

* .NET Framework 2.0 SP1
* Internet Explorer, не ниже 6.0
* Windows Installer 3.1

Поддерживаемые ОС: Windows XP SP2, SP3; Windows Vista; Windows 7; Windows 2003 Server; Windows 2008 Server.

Стоимость модуля электронного документооборота «M.E.Doc» сроком действия в 1 год для юридического лица – 2000 грн.

Из недостатков программы следует отметить то, что для малейшего расширения функционала пакета следует обращаться к техническому специалисту, например, для обеспечения возможности использования ключей ЭЦП «Masterkey».

Использование услуг специалистов также требует затрат, а каждый модуль «M.E.Doc» покупается отдельно, то есть, для приобретения нескольких модулей программы необходимо значительное количество средств.

Также со списка выделяется сервис разработанный компанией «iFin», он предназначен для удаленного ведения учета, автоматизированного формирования электронной отчетности и сдачи ее в налоговую инспекцию, пенсионный фонд и органы статистики. Сервис «iFin» позволяет не только создавать отчеты, но и импортировать их из других программ, таких как «1С: Бухгалтерия».

В отличии от «M.E.Doc», данный сервис не обладает модульной структурой. На выбор пользователя есть всего лишь 3 версии пакета с разным функциональным набором. Тем не менее, в сравнении с предыдущим программным пакетом, цены гораздо более демократичны: наиболее функциональная версия с лицензией на год стоит 1249 грн для юридических лиц.

Особенностью пакета является то, что он, по сути, не требует установки на компьютер. Доступ к данным можно получить с любого компьютера, главное – наличие стабильного подключения к интернету. Такой формат значительно снижает аппаратные требования для корректного функционирования.

Ввиду повышенного риска взлома и кражи данных, «iFin» использует защитный протокол «https», а также специальный протокол шифрования данных, который значительно снижает вероятность хищение личной информации пользователя. Несмотря на это, риск все же остается, поэтому, при всем удобстве сервиса, он не способен обеспечивать достаточный уровень безопасности данных.

Вышеупомянутые пакеты программного обеспечения пригодны для использования по назначению, а большинство из них находится в эксплуатации уже не один год. Тем не менее, рекомендуется отдать предпочтение системе, которая будет в состоянии обеспечить высокий уровень безопасности данных, так как их весомая часть тесно связана с финансовой деятельностью физических и юридических лиц.   
 Таблица 1.1 наглядно демонстрирует достоинства и недостатки рассмотренных систем.

Таблица 1.1 – Сравнение систем

|  |  |  |  |
| --- | --- | --- | --- |
| № | Характеристика | «M.E.Doc» | «iFin» |
| 1. | Низкие аппаратные требования | - | + |
| 2. | Низкие программные требования | - | + |
| 3. | Низкая цена | - | + |
| 4. | Высокая функциональность | + | - |

Продолжение таблицы 1.2 – Сравнение систем

|  |  |  |  |
| --- | --- | --- | --- |
| № | Характеристика | «M.E.Doc» | «iFin» |
| 5. | Гибкость и вариативность настроек | + | - |
| 6. | Высокий уровень безопасности данных | + | - |
| 7. | Универсальность применения | + | - |

* 1. **Обоснование необходимости совершенствования информационной системы**

Опираясь на данные о рассмотренных информационных системах, можно сказать, что, несмотря на предоставляемый широкий функционал, они обладают рядом недостатков, которые могут привести к снижению производительности той или иной структуры, которая будет эксплуатировать данную систему.

Рассматриваемые пакеты программного обеспечения позиционируются как системы электронной отчетности, соответственно, они не ориентированы на конкретную область деятельности предприятий, что значительно расширяет спектр их применения, однако из-за этого некоторые функции могут работать нестабильно. По сути, это ситуация, когда качество предоставляемых функций уступает их количеству.

Ожидается, что разрабатываемая подсистема не будет иметь столь широкий спектр применения, как ее аналоги, она будет ориентирована на конкретную сферу деятельности (например, налогообложение земельных участков). В результате удастся снизить системные требования, что очень кстати, так как большинство предприятий не обладает мощными компьютерами. Также из-за уменьшения количества функций интерфейс программы станет значительно проще и понятнее обычному пользователю, что фактически снимет вопрос о поиске высококвалифицированных кадров для эффективной эксплуатации программного пакета. Это положительно повлияет на производительность предприятия и снизит затраты.

* 1. **Назначение и цели создания подсистемы**

Разрабатываемая подсистема предназначена для расчета и учета налогообложенияземельных участков физическими и юридическими лицами.

Предполагается, что система будет использоваться отделом по отводу земли на предприятии и работниками районной налоговой инспекции.

Целью создания подсистемы является частичная автоматизация компьютеризация процесса учета налогообложения земельных участков, что должно привести к упрощению работы налоговых инспекторов и работников предприятия, деятельность которых связана с отводом земли.

Подсистема должна обеспечивать отправку налогоплательщиком и возможность проверки работником районной налоговой инспекцией пакета документов, который будет предоставлять всю необходимую информацию об объекте налогообложения. Исходя из данных, которые будут содержаться в пакете, разрабатываемая система должна предоставлять возможность рассчитать величину налога с учетом льгот и особенностей объекта налогообложения, определить наличие и величину просроченных налоговых обязательств, рассчитать пеню, недоимку и штраф.

Подсистема не должна обладать высокими требованиями к аппаратному обеспечению. Предполагаемые системные требования приведены в таблице 1.2.

Таблица 1.2 – Аппаратные требования разрабатываемой подсистемы

|  |  |  |
| --- | --- | --- |
| № | Характеристика | Значение |
| 1 | Частота процессора | Не менее 1,5 ГГц |
| 2 | Место на жестком диске | Не менее 200 Мб |
| 3 | Объем ОЗУ | Не менее 1 ГБ |

Продолжение таблицы 1.2 – Аппаратные требования разрабатываемой подсистемы

|  |  |  |
| --- | --- | --- |
| № | Характеристика | Значение |
| 4 | Сеть | От 10 Мбит/с |
| 5 | Операционная система | Windows XP или более поздняя версия |

Также рабочая станция должна быть оснащена манипулятором типа мышь, клавиатурой, струйным или лазерным принтером и цветным графическим дисплеем.

Эффективность работы системы можно будет определить по времени и усилиям, которые затрачиваются работником предприятия или районной налоговой инспекции в процессе ее эксплуатации. Затем, полученные показатели следует сравнить с аналогичными, полученными при работе с другими программными пакетами. По возможности можно также собрать данные, полученные путем анализа «фидбэка» (отзывов пользователей). Такой подход позволит получить более наглядную картину проблем и недочетов в программе. В будущем, основываясь на собранной информации, можно провести некоторые усовершенствования и исправления разработанной системы. Таким образом, в большей степени будут учтены пожелания обычных пользователей при ее разработке и совершенствовании.

Предполагается, что данный пакет программного обеспечения не предназначен для продажи в связи со спецификой объекта компьютеризации.

будет распространяться бесплатнокому он нужен!!.

**2 ОПРЕДЕЛЕНИЕ ФУНКЦИОНАЛЬНОЙ СТРУКТУРЫ ПРОЕКТИРОВАНИЯ КОМПЬЮТЕРИЗИРОВАННОЙ ПОДСИСТЕМЫ**

**2.1 Краткое описание выбранного CASE – средства CA All Fusion Process Modeler (BPWin) используемого для схематического представления проектируемой системы.**

CASE (Computer-Aided Software Engineering) — набор инструментов и методов программной инженерии для проектирования программного обеспечения, который помогает обеспечить высокое качество программ, отсутствие ошибок и простоту в обслуживании программных продуктов. Также под CASE понимают совокупность методов и средств проектирования информационных систем с использованием CASE-инструментов.

Computer Associates All Fusion Process Modeler (BPWin)  — [CASE](https://ru.wikipedia.org/wiki/CASE)-средство для [проектирования](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%B1%D0%B0%D0%B7_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) и [документирования](https://ru.wikipedia.org/wiki/%D0%94%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%B0%D1%86%D0%B8%D1%8F) [баз данных](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), которое позволяет создавать, документировать и сопровождать базы и хранилища данных. [Модели данных](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) помогают визуализировать структуру данных, обеспечивая эффективный процесс организации, управления и администрирования таких аспектов деятельности предприятия, как уровень сложности данных и технологий баз данных.

All Fusion Process Modeler (BPWin) предназначен для всех компаний, разрабатывающих и использующих базы данных, для [администраторов баз данных](https://ru.wikipedia.org/wiki/%D0%90%D0%B4%D0%BC%D0%B8%D0%BD%D0%B8%D1%81%D1%82%D1%80%D0%B0%D1%82%D0%BE%D1%80_%D0%B1%D0%B0%D0%B7_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), [системных аналитиков](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%BD%D1%8B%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D1%82%D0%B8%D0%BA), проектировщиков баз данных, [разработчиков](https://ru.wikipedia.org/w/index.php?title=%D0%A0%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D1%87%D0%B8%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F&action=edit&redlink=1), руководителей проектов. All Fusion Process Modeler позволяет управлять данными в процессе корпоративных изменений, а также в условиях стремительно изменяющихся технологий.

All Fusion Process Modeler позволяет наглядно отображать сложные структуры данных. Удобная в использовании графическая среда All Fusion Process Modeler упрощает разработку базы данных и автоматизирует множество трудоёмких задач, уменьшая сроки создания высококачественных и высокопроизводительных баз данных и хранилищ данных. Данное решение улучшает коммуникацию организации, обеспечивая совместную работу администраторов и разработчиков баз данных, многократное использование модели, а также наглядное представление комплексных данных в удобном для понимания и обслуживания формате.[где ссылка?сам придумал?]

**2.2 Описание функционально – структурной схемы проектируемой системы**

Целью проектирования является автоматизированная система учета земельных налогов в рамках районной налоговой инспекции для физических и юридических лиц. Начальная схема системы изображена на рисунке 2.1.

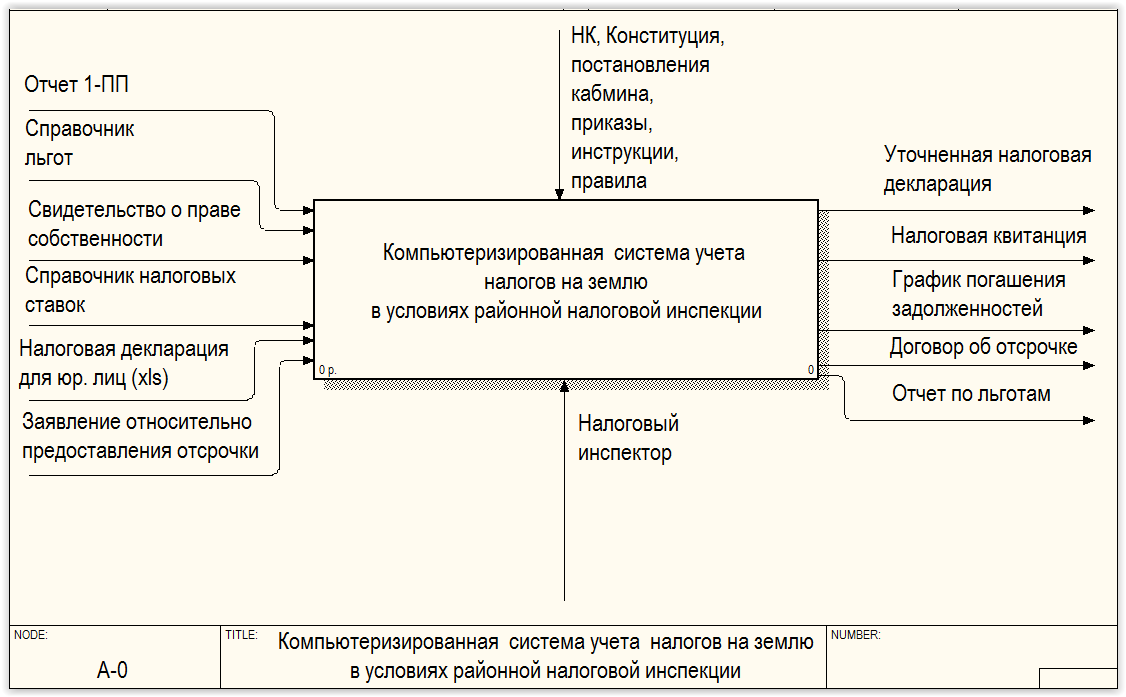


Рисунок 2.1 – Автоматизированная система учета налогов на земля

Как часто? Для формирования конечного расчета налога необходимо подать в налоговую инспекцию пакет документов, который будет содержать информацию о налогоплательщике и объекте налогообложения, а также правоустанавливающие документы на владение или использование земельного участка. Как правило, это документы подаются в электронном виде при помощи соответствующего программного обеспечения. Ярким примером такой программы является «Бест – звіт» (сейчас – «M.E.Doc»).

Основная часть работы налоговой инспекции выполняется налоговыми инспекторами, специализирующимися на работе с юридическими или физическими лицами.

Результатом выполнения функций налоговой инспекции станет расчет налога с учетом предоставляемых льгот и штрафов за просроченные налоговые обязательства, при условии их наличия.

**2.3 Определение функций проектируемой системы**

Автоматизированная система учета налогов на землю в рамках районной налоговой инспекции функционирует, взаимодействуя с персоналом и данными налоговой.

Ключевые функций автоматизируемой системы:

* Регистрация и учет налогоплательщиков;
* Учет налогообложения юридических лиц;
* Учет погашения просроченных налоговых обязательств;
* Формирование итоговых отчетов в органы статистики и вышестоящие инстанции;

Схема функционала системы автоматизируемой приведена на рисунке 2.2.

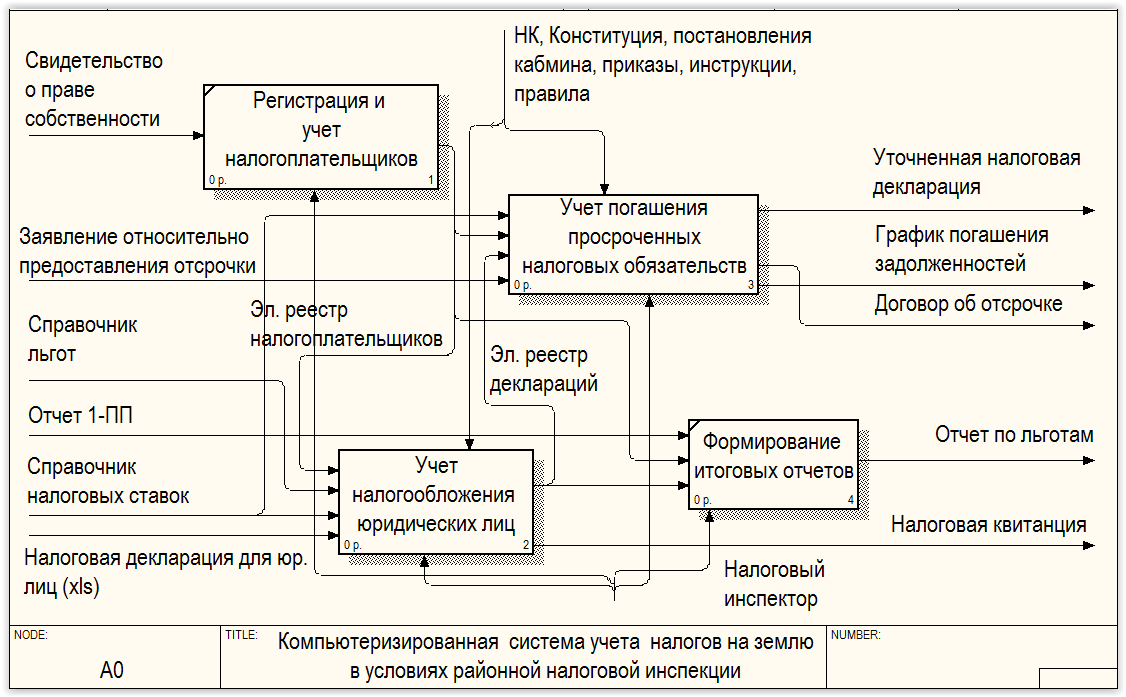


Рисунок 2.2 – Функционал автоматизируемой системы

При направлении в налоговую инспекцию, документация с информацией о налогоплательщике заносится в электронный реестр, где происходит проверка соответствия с уже имеющимися данными или же регистрация нового плательщика налогов, в случае отсутствия информации о нем в базе.

Для случая с просроченными налоговыми обязательствами у плательщика предусмотрена функция погашения долгов, которая также подразумевает формирование уточненной налоговой декларации и календарного плана уплаты просроченных платежей.

Для стандартных случаев предназначена функция учета налогообложения юридических лиц, которая на основании данных из реестра и декларации формирует налоговую квитанцию.

**2.4 Описание функции «Учет налогообложения юридических лиц»**

Функция «Учет налогообложения юридических лиц» в виде «черного ящика» приведена на рисунке 2.3.

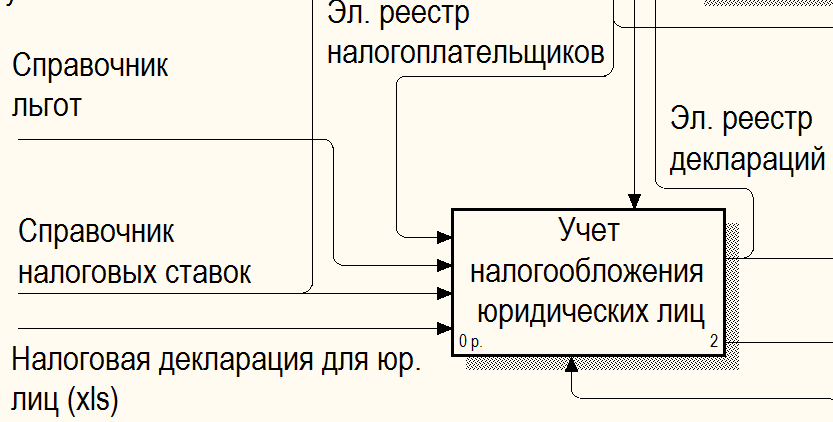


Рисунок 2.3 – «Черный ящик» функции «Учет налогообложения юридических лиц»

Функция формирует налоговую квитанцию, оперируя данными из электронного реестра плательщиков и декларации юридического лица.

Для более детального рассмотрения функции следует провести ее декомпозицию, схема полученная в результате приведена на рисунке 2.4.

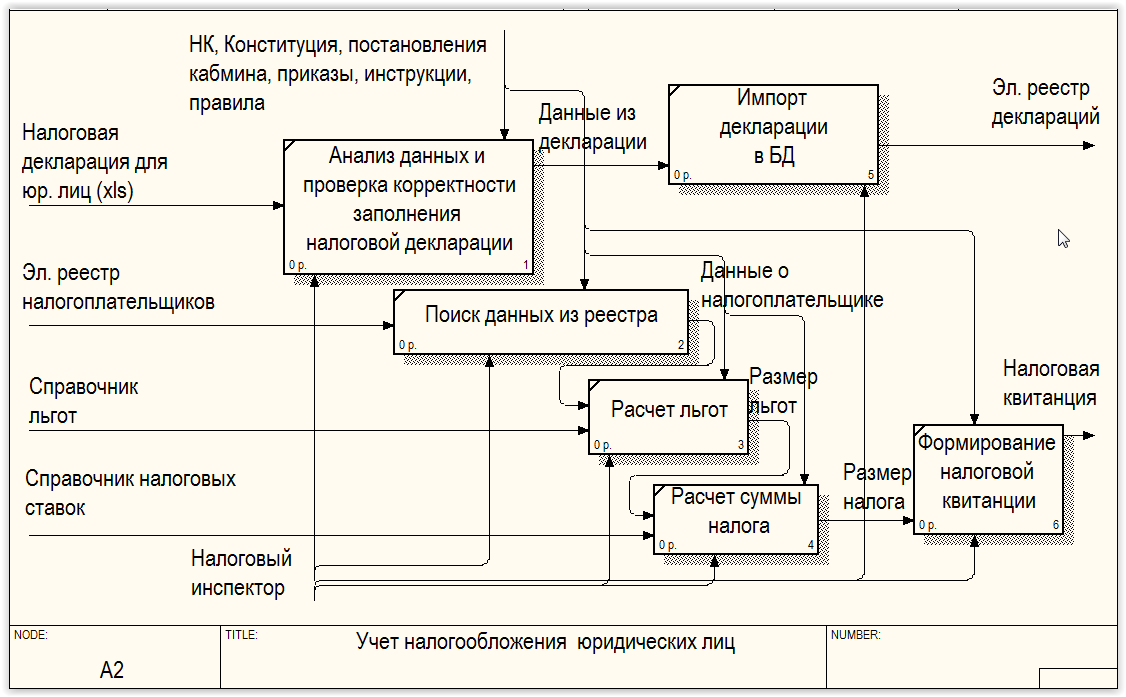


Рисунок 2.4 – Декомпозиция функции «Учет налогообложения юридических лиц»

Отправленная юридическим лицом налоговая декларация проходит проверку на корректность заполнения, данные из нее заносятся в базу. На основании этих данных и информации из электронного реестра налогоплательщиков происходит расчет льгот и конечной суммы налога. Информация о предоставляемых плательщику льготах и величина налоговой ставки определяются специальными справочниками.

**2.5 Описание функции «Учет погашения просроченных налоговых обязательств»**

Функция «Учет погашения просроченных налоговых обязательств» в виде «черного ящика» представлена на рисунке 2.5.

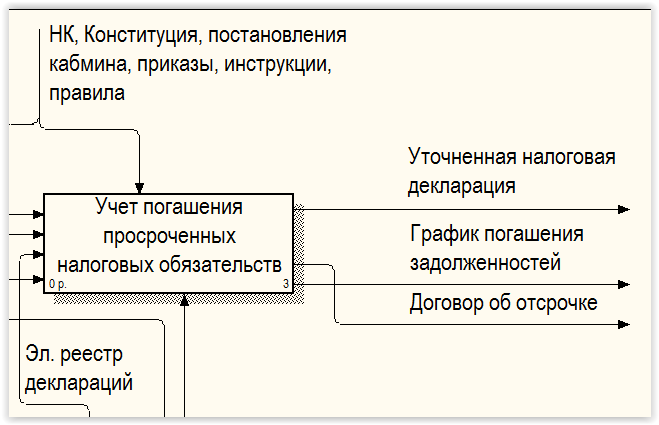


Рисунок 2.5 - «Черный ящик» функции «Учет погашения просроченных налоговых обязательств»

При наличии у налогоплательщика задолженностей будет сформирована уточненная декларация. Заявление на отсрочку в случае его подачи рассматривается, принимается соответствующее решение на предоставление отсрочки и на его основании формируется договор об отсрочке.

Для более детального анализа функции погашения налоговых обязательств проведем ее декомпозицию, результаты которой приведены на рисунке 2.6.



Рисунок 2.6 – Декомпозиция функции «Учет погашения просроченных налоговых обязательств»

На основании анализа налоговой декларации и информации из реестра определяется наличие просроченных налоговых обязательств у плательщика.

В случае наличия задолженностей происходит расчет недоимки, штрафов и пени. На основании этих расчетов формируется уточненная декларация по налоговым обязательствам.

**2.6 Определение функциональной взаимосвязи компьютеризи-рованной подсистемы с другими подсистемами, функционирующими на предприятии**

Работа компьютеризированной системы учета платежей земельного налога тесно связана с налоговой системой государства, его правовым полем, а также деятельностью юридических лиц.

Система оперирует данными, которые отправляются юридическим лицом, заносит их в базу, формирует на основе этих данных некоторые документы (квитанция, уточненная декларация, договор об отсрочке) и отправляет их обратно налогоплательщику.

Справочная информация, используемая в расчетах, хранится на серверах государственной налоговой инспекции.

На основе данных за период формируются отчеты, которые предназначены для отправки в вышестоящие инстанции.

**3 РАЗРАБОТКА ИНФОРМАЦИОННОГО ОБЕСПЕЧЕНИЯ**

**3.1 Выбор средства управления данными**

Все данные будут хранится на сервере районной налоговой инспекции, поэтому разрабатываемая информационная база будет однородной и централизованной, построенной по принципу «клиент – серверной» архитектуры.

Клиент – сервер – это вычислительная архитектура, при которой нагрузка распределена между поставщиками услуг (серверами) и заказчиками (клиентами). [3]

Благодаря тому, что основная масса вычислений производится сервером, отпадает необходимость в мощных клиентских станциях. При этом также увеличивается уровень безопасности данных, так как сервер защищен гораздо лучше большинства клиентов, более того, там гораздо легче контролировать предоставляемый уровень полномочий при работе с базой данных.

Тем не менее, использование такой архитектуры обладает рядом недостатков:

* Проблемы на стороне сервера грозят неработоспособностью всей системы;
* Для поддержки рабочего состояния системы необходим отдельный специалист;
* Приобретение специального оборудования требует затрат (однако, имеется возможность сэкономить на мощностях клиентских станций).

Выбор системы управления базой данных осуществляется на основе модели организации данных, а также уже имеющегося опыта работы. При выборе были рассмотрены следующие современные системы: MySQL, MS SQL Server, Oracle.

В таблице 3.1 приведены характеристики двух СУБД, выбор которых более предпочтителен. Сравнение производилось между версиями Oracle Database SE 11g и Microsoft SQL Server 2008 SE R2 так как обе системы являются корпоративными и принадлежат примерно к одной ценовой категории.

Таблица 3.1 – Характеристики СУБД Oracle и MS SQL Server

|  |  |  |  |
| --- | --- | --- | --- |
| № | Название характеристики | Oracle | MS SQL Server |
| 1. | Поддержка различных операционных систем | Windows, Linux, Unix, Mac OS, Solaris | Windows |
| 2. | Поддержка различных кодировок | Есть | Только Windows и Unicode |
| 3. | Поддержка нескольких экземпляров БД | Неограниченно | 16 |
| 4. | Минимальное количество пользовательских лицензий | 5 | 5 |
| 5. | Рекомендованная стоимость 5 пользовательских лицензий | 5 x 350 = 1750 $ | 950 + 5 x 175 = 1825 $ |
| 6. | Переход на новые версии | Бесплатно при наличии технической поддержки | Полная стоимость новой версии |
| 7. | Доступность технической документации | Только для обладателей | Свободный доступ |
| 8. | Поддержка технологии PowerShell | Нет | Есть |
| 9. | Легкая интеграция с продуктами Microsoft | Нет | Есть |

Данные из таблицы 3.1 говорят о том, что, СУБД Oracle более универсальна и поддерживает множество операционных систем, при этом более демократична в плане стоимости, особенно при переходе на новую версию, тем не менее MS SQL Server больше подходит для работы с базами данных на платформе Windows. Выбору продукта от Microsoft также способствует наличие информативной и понятной документации в свободном доступе, а совместимость с другими программными пакетами от компании сделают эксплуатацию SQL Server гораздо комфортнее.

С учетом требований к информационному обеспечению, для управления базой данных была выбрана система реляционного типа Microsoft SQL Server 2014.

Выбранная СУБД обладает рядом достоинств при разработке информационного обеспечения:

* Простота использования. Легкость в эксплуатации позволяет снизить операционные издержки и расходы на разработку решений на основе платформы;
* Управляемость. Интуитивно понятные средства управления и автоматизированного администрирования помогают эффективно управлять бизнес-приложениями;
* Создание отчетов и аналитика. С помощью встроенных технологий анализа и создания отчетности можно легко и быстро получать практическую и значимую информацию, а на ее основании принимать решения;
* Интеграция с другими продуктами Microsoft, такими как Excel и Access;
* Поддержка со стороны производителя, постоянный выход обновлений;
* Огромная распространенность, как следствие – большое количество справочного материала, документации и технических инструкций.

**3.2 Описание классификаций системы кодирования**

При разработке информационной модели для кодирования реквизитов используются общепринятые системы классификации и кодирования. Основным источником кодировок выступают действующие государственные системы и отраслевые инструкции. Структура кодовых обозначений объектов приведена в таблице 3.2.

Таблица 3.2 – Структура классификаторов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Значность кода | Система классификации | Система кодирования | Вид классификатора |
| КОАТУУ | 10 | многоаспектная | комбинационная | государственный |
| КВЭД | 6 | иерархическая | комбинационная | государственный |
| Код ЕГРПОУ | 8 | многоаспектная | комбинационная | государственный |
| Код категории земли | 2 | отсутствует | порядковая | отраслевой |
| Код вида права собственности | 1 | отсутствует | порядковая | отраслевой |
| Кадастровый номер | 19 | многоаспектная | комбинационная | отраслевой |

Продолжение таблицы 3.2 – Структура классификаторов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Значность кода | Система классификации | Система кодирования | Вид классификатора |
| Серия паспорта | 2 | многоаспектная | комбинационная | государственный |
| Номер паспорта | 6 | многоаспектная | комбинационная | государственный |
| Почтовый индекс | 5 | иерархический | комбинационная | государственный |

**3.3 Разработка модели данных**

Логическая и физическая модели данных разрабатываются на основе детального анализа предметной области, в ходе которого определяются основные сущности системы и связи между ними. Эти модели используются для создания базы данных как основы информационного обеспечения подсистемы.

В основе проектирования информационных моделей лежит принцип разбиения структуры входных документов на таблицы, каждой из которых соответствует одна сущность.

Перечень сущностей и их описание приведены в таблице 3.3.

Таблица 3.3 – Сущности модели компьютеризированной подсистемы начисления и учета платежей земельного налога в условиях районной налоговой инспекции

|  |  |  |
| --- | --- | --- |
| № | Сущность | Описание |
| 1. | Юридическое лицо | Основная информация о налогоплательщике, предоставляемая в налоговой декларации. |
| 2. | Декларация | Информация о типе декларации и периоде налогообложения. |
| 3. | Земельный участок | Данные об объекте налогообложения. |
| 4. | Контактные данные юр. лица | Адресные и контактные данные налогоплательщика. |
| 5. | Налоговые обязательства | Данные о суммах налога, штрафов и пени. |
| 6. | Свидетельство о праве собственности | Информация о зарегистрированном праве владения земельным участком. |
| 7. | Льготы | Данные о предоставляемых льготах. |
| 8. | Отчет 1 – ПП | Отчет о суммах льгот в налогообложении юридических лиц. |
| 9. | Заявление об отсрочке | Заявление на предоставление отсрочки налоговых обязательств юридического лица. |
| 10. | Решение об отсрочке | Принятое налоговым органом решение о предоставлении отсрочки. |
| 11. | Журнал регистрации отсрочек | Информация о принятых решениях. |

Атрибуты сущностей модели приведены в таблице 3.4.

Таблица 3.4 – Атрибуты сущностей модели

|  |  |  |
| --- | --- | --- |
| № | Сущность | Атрибуты |
| 1. | Юридическое лицо | Код юр. лица, наименование (полное), наименование (краткое), ФИО руководителя, ФИО главного бухгалтера, код контактных данных, ид. код по ЕГРПОУ, КВЭД, КОАТУУ, контролирующий орган, рег. номер учетной карточки налогоплательщика / серия и номер паспорта руководителя, рег. номер учетной карточки налогоплательщика / серия и номер паспорта главного бухгалтера. |
| 2. | Декларация | Код декларации, тип декларации, год, месяц, начало периода, конец периода, дата заполнения, ФИО заполнителя, дата внесения в базу, код юр. лица, код участка, код обязательств. |
| 3. | Земельный участок | Код участка, категория земли, код свидетельства, НДО. ед. площади зем. участка, ставка налога, годовая сумма земельного налога, код льготы, годовая сумма земельного налога (к оплате). |
| 4. | Льготы | Код льготы, код справочника льгот, размер льготы, сумма льготы. |
| 5. | Налоговые обязательства | Код обязательства, год, насчитано по данным ранее поданной налоговой декларации, насчитано всего, насчитано до увеличения налогового обязательства, насчитано до уменьшения налогового обязательства, размер уменьшения налогового обязательства, размер увеличения налогового обязательства, сумма штрафа, сумма пени. |

Продолжение таблицы 3.5 – Атрибуты сущностей модели

|  |  |  |
| --- | --- | --- |
| № | Сущность | Атрибуты |
| 6. | Контактные данные юр. лица | Код контактных данных, почтовый индекс, межгородской код, телефон, факс, электронный адрес. |
| 7. | Заявление на отсрочку | Код заявления на отсрочку, код юр. лица, тип заявления, наименование долга, сумма отсрочки, начало нового периода, конец нового периода, пункт перечня обстоятельств, раздел перечня обстоятельств. |
| 8. | Решение об отсрочке | Код решения об отсрочке, код заявления на отсрочку, начало установленного периода, конец установленного периода, установленный срок, проценты, сумма уплаты, отменено. |
| 9. | Журнал регистрации отсрочек | Код журнала, код регистрации, дата принятия решения, код договора, долг уплачен. |
| 10. | Свидетельство о праве собственности | Код свидетельства, адрес земельного участка, кадастровый номер, вид права собственности, номер документа, серия документа, целевое предназначение земельного участка, площадь земельного участка, дата регистрации, ФИО регистратора, орган регистрации. |
| 11. | Отчет 1 – ПП | Код отчета 1 – ПП, код юр. лица, код суммы льгот в налогообложении, код суммы использованных льгот, дата начала действия, дата окончания действия, всего льгот в налогообложении, льгот из бюджета, использовано льгот, использовано льгот из бюджета. |

Разработка моделей данных осуществляется при помощи CASE – средства MS Visio 2013.

При выборе между средствами Erwin DataModeler и MS Visio последний показал себе с лучшей стороны в плане удобства использования, а также простоте построения моделей. Атрибуты и их тип разделены на отдельные колонки, внешние и первичные ключи выделены специальным значком, существует возможность изменения стиля и количества отображаемой информации – все это делает использование Visio гораздо удобнее и приятнее, а интуитивно понятный интерфейс обеспечит комфортную и продуктивную работу с уже готовой моделью.

При работе со средством Erwin следует отметить визуальную компактность сущностей и минималистический интерфейс, что делает построение модели несколько легче. Однако, тип и размер атрибутов не выносятся в отдельную колонку, вместо этого вся информация о поле пишется одной цельной строкой – это значительно понижает степень наглядности информации, затрудняя беглый просмотр данных модели.

При разработке модели определяются сущности, их ключи, атрибуты, а также связи между сущностями. Процедура нормализации обеспечивает приведение таблиц базы данных к нормальным формам, устраняя частичные и транзитивные зависимости неключевых полей от ключа. Логическая модель данных изображена на рисунке 3.1.

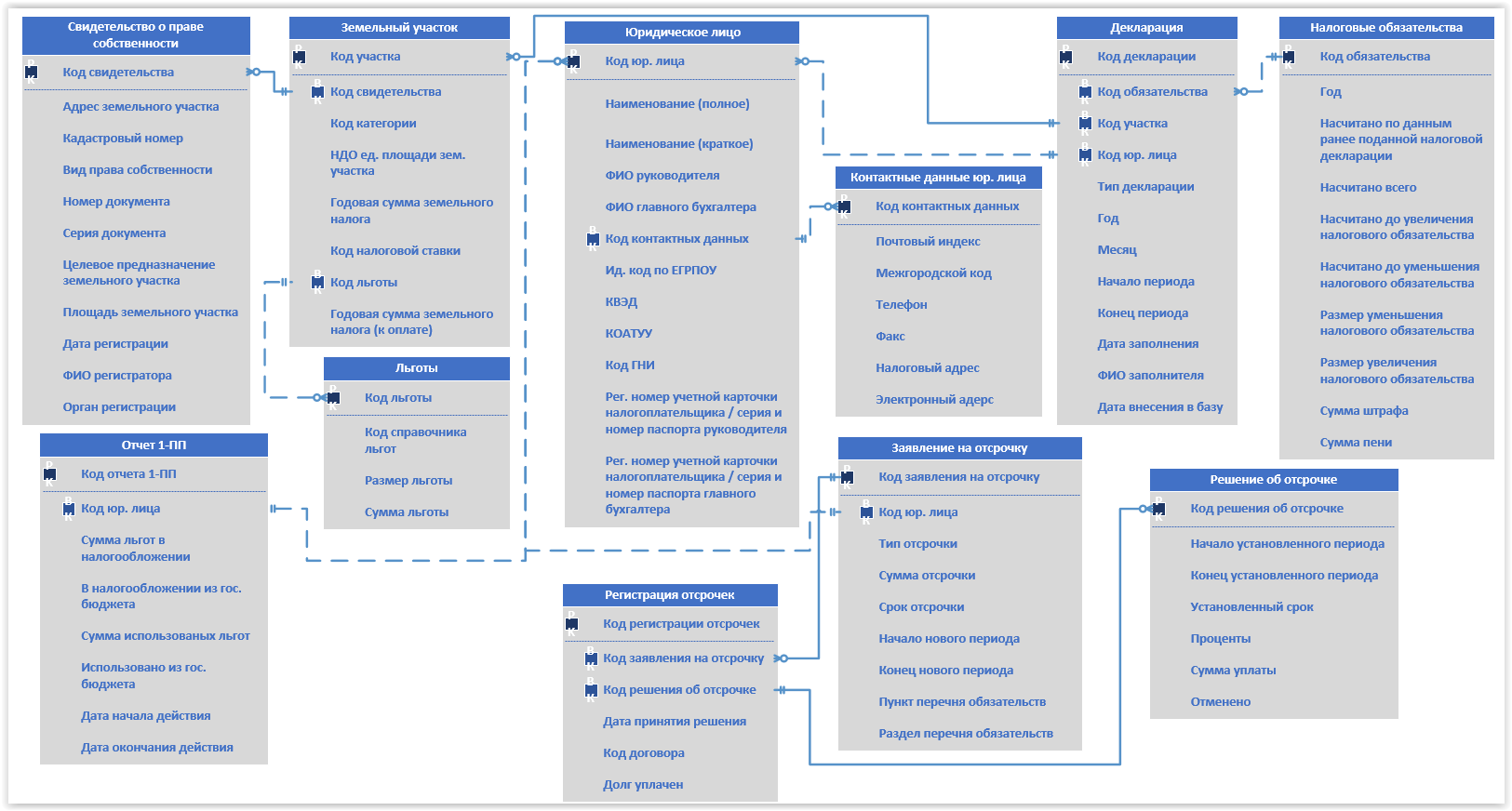


Рисунок 3.1 – ЛМД

Переход от логической модели к физической обеспечивается заменой сущностей на таблицы, а атрибутов – на поля, для которых определяется тип и размер.

Описание физической модели трех основных сущностей приведено в таблицах 3.5 – 3.7.

Таблица 3.5 – Структура таблицы «Юридическое лицо»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя поля | Тип | Размер | Длина, б |
| 1. | Код юр. лица | Int | 3 | 12 |
| 2. | Наименование (полное) | Varchar | 100 | 100 |
| 3. | Наименование (краткое) | Varchar | 30 | 30 |
| 4. | ФИО руководителя | Varchar | 45 | 45 |
| 5. | ФИО главного бухгалтера | Varchar | 45 | 45 |
| 6. | Код контактных данных | Int | 3 | 12 |
| 7. | Ид. код по ЕГРПОУ | Varchar | 8 | 8 |
| 8. | КОАТУУ | Varchar | 10 | 10 |
| 9. | КВЭД | Varchar | 7 | 7 |
| 10. | Код ГНИ | Int | 3 | 12 |
| 11. | Рег. номер учетной карточки налогоплательщика / серия и номер паспорта руководителя | Varchar | 10 | 10 |
| . | Рег. номер учетной карточки налогоплательщика / серия и номер паспорта главного бухгалтера | Varchar | 10 | 10 |
| Всего | | | 274 | 301 |

Таблица 3.6 – Структура таблицы «Земельный участок»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя поля | Тип | Размер | Длина, б |
| 1. | Код участка | Int | 3 | 12 |
| 2. | Код категории | Int | 2 | 8 |
| 3. | Код свидетельства | Int | 3 | 12 |
| 4. | НДО ед. площади земельного участка | Float | 5 | 20 |
| 5. | Код налоговой ставки | Int | 4 | 16 |
| 6. | Годовая сумма земельного налога | Money | 9 | 72 |
| 7. | Код льготы | Int | 3 | 12 |
| 8. | Годовая сумма земельного налога (к оплате | Money | 9 | 72 |
| Всего | | | 38 | 224 |

Таблица 3.7 – Структура таблицы «Налоговые обязательства»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя поля | Тип | Размер | Длина, б |
| 1. | Код обязательства | Int | 3 | 12 |
| 2. | Год | Date | 4 | 16 |
| 3. | Насчитано по данным ранее поданной налоговой декларации | Money | 8 | 64 |
| 4. | Насчитано всего | Money | 8 | 64 |
| 5. | Насчитано до увеличения налогового обязательства | Money | 8 | 64 |

Продолжение таблицы 3.7 – Структура таблицы «Налоговые обязательства»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя поля | Тип | Размер | Длина, б |
| 6. | Насчитано до уменьшения налогового обязательства | Money | 8 | 64 |
| 7. | Размер уменьшения налогового обязательства | Money | 6 | 48 |
|  | Размер увеличения налогового обязательства | Money | 6 | 48 |
| 8. | Сумма штрафа | Money | 8 | 64 |
| 9. | Сумма пени | Money | 8 | 64 |
| Всего | | | 67 | 508 |

В процессе перехода были выделены новые таблицы…?

Вид физической модели данных приведен на рисунке 3.2.

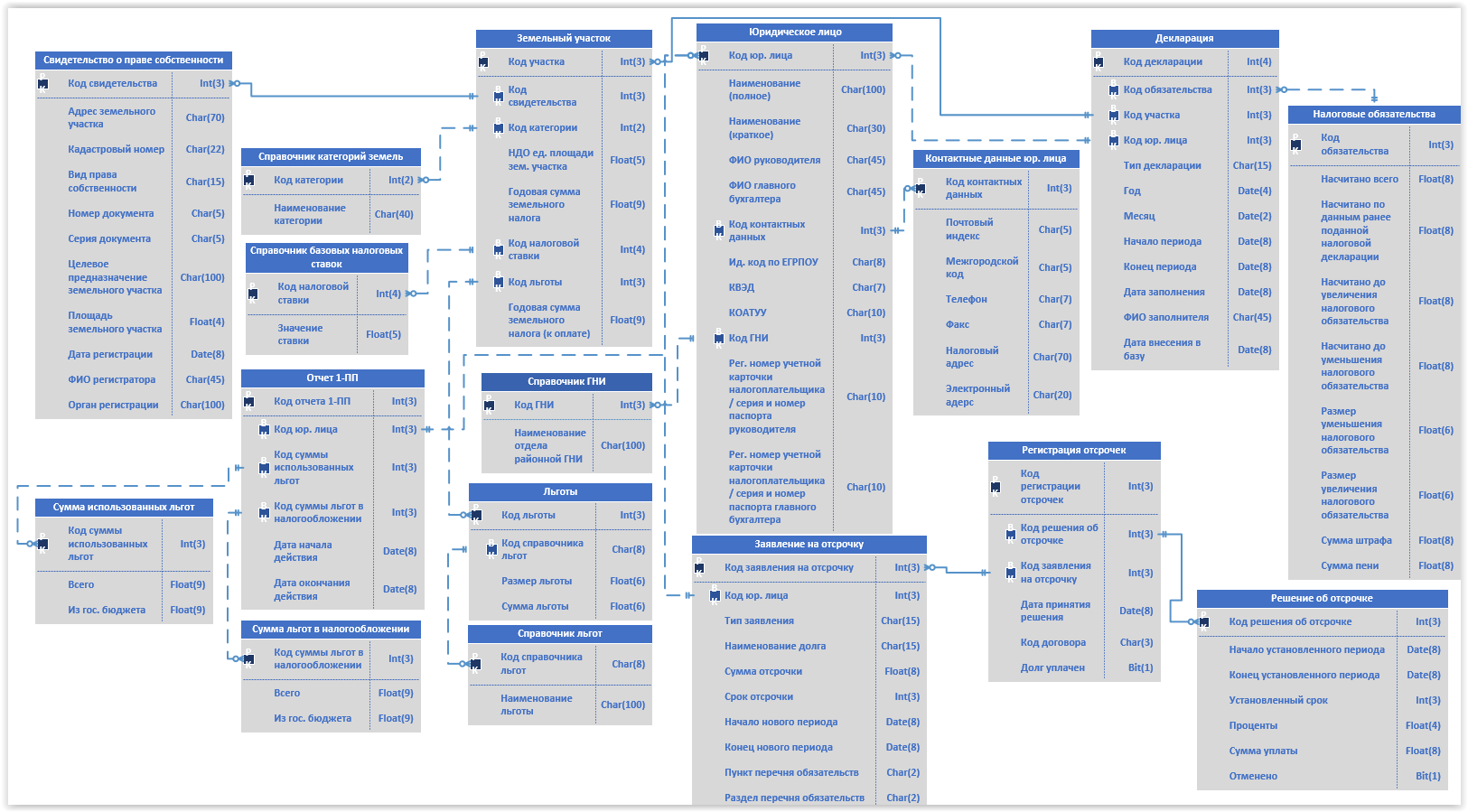


Рисунок 3.2 – ФМД

**3.4 Реализация базы данных**

Создание спроектированной базы происходит в среде разработки Microsoft SQL Server Management Studio, которая предназначена для конфигурирования, управления и администрирования компонентов SQL Server.

Проектирование и реализация интерфейса, а также его атрибутов (масок ввода, проверки вводимых значений и т. д.) происходит на этапе разработки клиентского приложения средствами среды MS Visual Studio 2013.

Описание реализации трех основных объектов базы данных приведены в таблицах 3.8 – 3.10.

Таблица 3.8 – Основные свойства полей таблицы «Юридическое лицо»

|  |  |  |
| --- | --- | --- |
| Поле | Подпись | Может быть NULL |
| Kod\_lica  (autoinc) | Код юр. лица |  |
| Name\_full | Наименование (полное) | + |
| Name\_short | Наименование (краткое) | + |
| Boss\_fio | ФИО руководителя |  |
| Buh\_fio | ФИО главного бухгалтера | + |
| Kod\_kont | Код контактных данных |  |
| Egrpou | Ид. код по ЕГРПОУ | + |
| Kved | КВЭД |  |
| Koatuu | КОАТУУ |  |
| Kod\_gni | Код ГНИ |  |

Продолжение таблицы 3.8 – Основные свойства полей таблицы «Юридическое лицо»

|  |  |  |
| --- | --- | --- |
| Поле | Подпись | Может быть NULL |
| Boss\_reg\_num | Рег. номер учетной карточки /серия и номер паспорта руководителя |  |
| Buh\_reg\_num | Рег. номер учетной карточки /серия и номер паспорта главного бухгалтера | + |

Таблица 3.9 – Основные свойства полей таблицы «Земельный участок»

|  |  |  |  |
| --- | --- | --- | --- |
| Поле | Подпись | Может быть NULL | Проверка значения |
| Kod\_uch(autoinc) | Код участка |  |  |
| Kod\_kat | Код категории |  |  |
| Kod\_svd | Код свидетельства |  |  |
| Ndo | НДО ед. площади зем. участка |  | >0 |
| Kod\_stavki | Код налоговой ставки |  |  |
| God\_sum | Годовая сумма земельного налога |  | >0 |
| Kod\_lgot | Код льготы | + |  |
| God\_sum\_pay | Годовая сумма земельного налога (к оплате) |  | >0 |

Таблица 3.10 – Основные свойства полей таблицы «Налоговые обязательства»

|  |  |  |  |
| --- | --- | --- | --- |
| Поле | Подпись | Может быть NULL | Проверка значения |
| Kod\_obz(autoinc) | Код обязательства |  |  |
| God | Год |  | < Текущий год |
| Count\_pred\_dekl | Насчитано по данным ранее поданной налоговой декларации |  | >0 |
| Count\_all | Насчитано всего |  | >0 |
| Count\_do\_uvel | Насчитано до увеличения налогового обязательства | + | >0 |
| Count\_do\_umen | Насчитано до уменьшения налогового обязательства | + | >0 |
| Razm\_umensh | Размер уменьшения налогового обязательства | + | >0 |
| Razm\_uvel | Размер увеличения налогового обязательства | + | >0 |
| Shtraf\_sum | Сумма штрафа | + | >0 |
| Penya\_sum | Сумма пени | + | >0 |

Перечень и назначение остальных объектов базы приведены в таблице 3.11.

Таблица 3.11 – Объекты БД

|  |  |  |
| --- | --- | --- |
| № пп | Имя объекта | Назначение объекта |
| Запросы | | |
| 1. | зНалоговая\_квитанция | Запрос на данные по уплате, в случае отсутствия просроченных обязательств. |

Продолжение таблицы 3.11 – Объекты БД

|  |  |  |
| --- | --- | --- |
| № пп | Имя объекта | Назначение объекта |
| 2. | зУточненная\_налоговая\_ декларация | Запрос на данные для формирования уточненной декларации, в случае наличия просроченных обязательств. |
| 3. | зДанные\_об\_отсрочке | Запрос на данные о решении по предоставлению отсрочки для формирования договора. |
| 4. | зОтчет\_по\_льготам | Запрос на данные для формирования отчета по льготам. |
| Процедуры | | |
| 1. | пОчистка\_Юр\_лица | Очистка данных с учетом обнуления автоматически инкрементируемых индексов каскада таблиц «Юр. лицо – Справочник ГНИ – Контактные данные». |
| 2. | пОчистка\_1ПП | Очистка таблиц «Сумма льгот – Сумма используемых льгот – Отчет 1 – ПП». |
| 3. | пОчистка\_Зем\_участок | Очистка таблиц «Справочник льгот – Справочник категорий – Свидетельство о праве собственности – Льготы – Земельный участок». |
| 4. | пОчистка\_Отсрочки | Очистка таблиц «Заявление на отсрочку – Решение об отсрочке – Регистрация отсрочек». |
| 5. | пОчистка\_Декларация | Очистка таблиц «Декларация – Налоговые обязательства». |
| 6. | пЗаполнение\_Юр\_лицо | Заполнение таблиц «Юр. лицо – Справочник ГНИ – Контактные данные». |

Продолжение таблицы 3.11 – Объекты БД

|  |  |  |
| --- | --- | --- |
| № пп | Имя объекта | Назначение объекта |
| 6. | пЗаполнение\_Юр\_лицо | Заполнение таблиц «Юр. лицо – Справочник ГНИ – Контактные данные». |
| 7. | пЗаполнение\_1ПП | Заполнение таблиц «Сумма льгот – Сумма используемых льгот – Отчет 1 – ПП». |
| 8. | пЗаполнение\_Зем\_участок | Заполнение таблиц «Справочник льгот – Справочник категорий – Свидетельство о праве собственности – Льготы – Земельный участок». |
| 9. | пЗаполнение\_Отсрочки | Заполнение таблиц «Заявление на отсрочку – Решение об отсрочке – Регистрация отсрочек». |
| 10. | пЗаполнение\_Декларации | Заполнение таблиц «Декларация – Налоговые обязательства». |

Схема полностью спроектированной базы данных в среде MS SQL Server Management Studio изображена на рисунке 3.3.

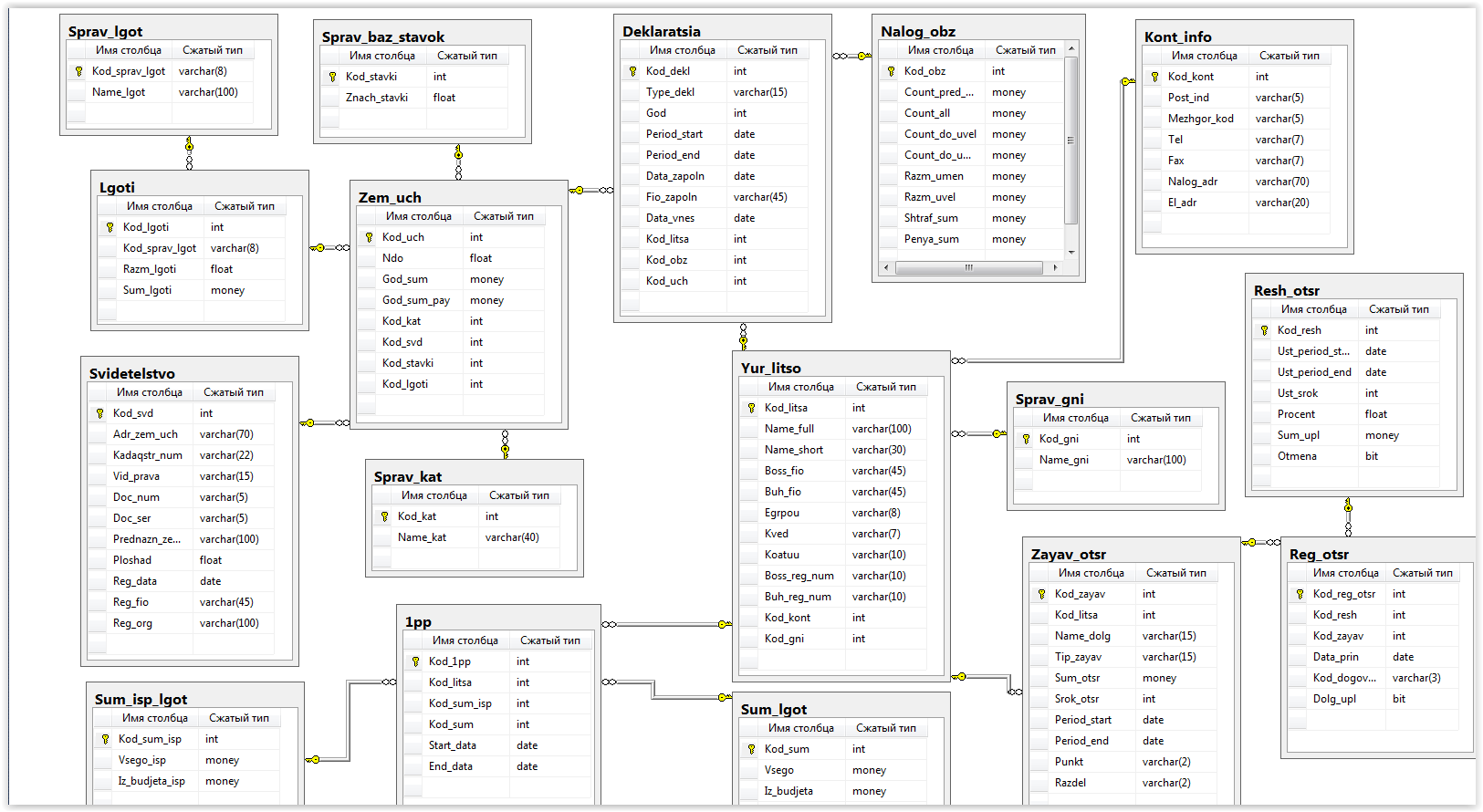


Рисунок 3.3 – Схема данных в среде разработки

**3.5 Организация сбора и обработки информации**

Расчет необходимого для хранения базы дискового пространства производится с учетом использования данных в течении пяти лет, при том условии если в районную налоговую инспекцию платят налог 50 предприятий, отсылая документы один раз за период длиною в год.

Расчет производился в килобайтах, опираясь на резервируемое средой пространство, результаты приведены в таблице 3.8.

Таблица 3.8 – Расчет объема БД

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название таблицы | Объем заголовка | Объем 1-й записи | Количество записей | Объем записей | Общий объем |
| Юридическое лицо | 8 | 2 | 50 | 100 | 108 |
| Декларация | 8 | 2 | 250 | 500 | 508 |
| Земельный участок | 8 | 2 | 250 | 500 | 508 |
| Льготы | 8 | 2 | 30 | 60 | 68 |
| Справочник льгот | 8 | 2 | 60 | 120 | 128 |
| Справочник ГНИ | 8 | 2 | 490 | 980 | 988 |
| Справочник базовых налоговых ставок | 8 | 2 | 1800 | 3600 | 3608 |
| Справочник категорий земель | 8 | 2 | 18 | 36 | 44 |
| Налоговые обязательства | 8 | 2 | 150 | 300 | 308 |
| Контактные данные юр. лица | 8 | 2 | 50 | 100 | 108 |

Продолжение таблицы 3.9 – Расчет объемов БД

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название таблицы | Объем заголовка | Объем 1-й записи | Количество записей | Объем записей | Общий объем |
| Заявление на отсрочку | 8 | 2 | 150 | 300 | 308 |
| Решение об отсрочке | 8 | 2 | 150 | 300 | 308 |
| Журнал регистрации отсрочек | 8 | 2 | 150 | 300 | 308 |
| Свидетельство о праве собственности | 8 | 2 | 250 | 500 | 508 |
| Отчет 1 – ПП | 8 | 2 | 250 | 500 | 508 |
| Сумма льгот в налогообложении | 8 | 2 | 250 | 500 | 508 |
| Сумма использованных льгот | 8 | 2 | 250 | 500 | 508 |
| Всего | | | | | 8140 |

Процесс начинается с приема налоговой декларации, которая проходит проверку на корректность заполнения. Данные из декларации импортируются в базу.

На данные таблиц «Юридическое лицо», «Декларация» и «Земельный участок» выполняется запрос «зНалоговая квитанция», который используется для формирования выходного документа – налоговой квитанции.

Отчет 1 – ПП создается на основе данных, получаемых посредством выполнения запроса «зОтчет по Льготам» на данные из таблиц «Отчет 1 – ПП», «Сумма льгот в налогообложении» и «Сумма использованных льгот».

Для формирования уточненной налоговой декларации выполняется запрос «зУточненная декларация» на данные из таблиц «Декларация», «Земельный участок», «Юридическое лицо» и «Налоговые обязательства».

Данные по выходным документам при их формировании выводятся на экран.

Схема сбора обработки и передачи информации изображена на рисунке 3.3.

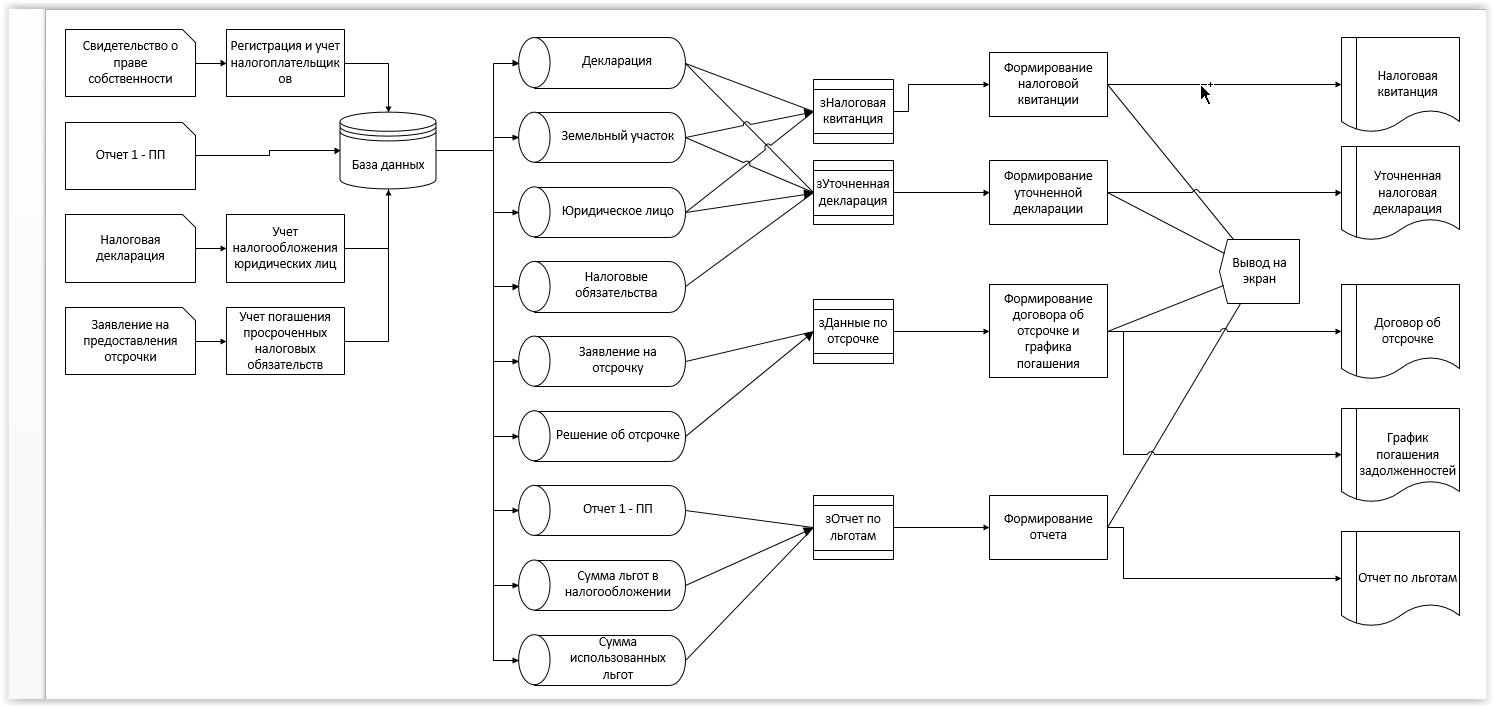


Рисунок 3.3 – Схема сбора обработки и передачи информации

**4 РАЗРАБОТКА МАТЕМАТИЧЕСКОГО И АЛГОРИТМИЧЕСКОГО ОБЕСПЕЧЕНИЯ**

**4.1 Описание алгоритма функции «Учет налогообложения юридических лиц»**

**4.1.1 Назначения и характеристики функции**

Функция предназначена для осуществления процедуры учета налогообложения земельного участка, собственником которого является юридическое лицо.

Пользователем программы, предоставляющей функционал, является налоговый инспектор, вся деятельность которого сводится к активации основных процессов (расчеты льгот и налога, импорт документов в базу данных и т.д.) путем нажатия на соответствующие клавиши.

Основным входным документом является налоговая декларация актуальной на 2016 год формы. Декларация должна быть с расширением «.xml», такой формат позволит с легкостью импортировать данные в базу.

Справочники с разнообразной информацией по налогообложению хранятся в виде таблиц базы данных.

Выходным документом является налоговая квитанция в текстовом формате с расширением «.doc». Офисный пакет от Microsoft установлен практически на каждой рабочей станции под управлением ОС Windows, что исключает невозможность ознакомления налогоплательщика с квитанцией.

**4.1.2 Используемая информация для функции**

Некоторые таблицы БД используются в качестве справочного материала при заполнении основных таблиц информацией из налоговой декларации.

Перечень используемых при реализации алгоритма «справочных» таблиц, приведен в таблице 4.1.

Таблица 4.1 – Используемые таблицы БД для функции «Учет налогообложения юридических лиц»

|  |  |  |
| --- | --- | --- |
| № | Наименование таблицы | Поля |
| 1. | Льготы | Код льгота, код справочника льгот, размер льготы, сумма льготы. |
| 2. | Справочник базовых налоговых ставок | Код налоговой ставки, значение ставки |
| 3. | Справочник ГНИ | Код ГНИ, наименование отдела районной ГНИ |
| 4. | Справочник категорий | Код категории, наименование категории |
| 5. | Свидетельство о праве собственности | Код свидетельства, адрес земельного участка, кадастровый номер, вид права собственности, номер документа, серия документа, целевое предназначение земельного участка, площадь земельного участка, дата регистрации, ФИО регистратора, орган регистрации. |

**4.1.3 Результаты работы функции**

Функция носит учетный характер, то есть ее цель – это не конкретные вычисления, а занесение в базу готовых данных, содержащихся во входных документах. Перечень объектов базы, использующихся для сохранения входной информации приведен в таблице 4.2.

Таблица 4.2 – Таблицы БД для сохранения результатов работы функции «Учет налогообложения юридических лиц»

|  |  |  |
| --- | --- | --- |
| № | Наименование таблицы | Поля |
| 1. | Декларация | Код декларации, тип декларации, год, месяц, начало периода, конец периода, дата заполнения, ФИО заполнителя, дата внесения в базу, код участка, код юр. лица |
| 2. | Юридическое лицо | Код юр. лица, наименование (полное), наименование (краткое), ФИО руководителя, ФИО глав. бухгалтера, код контактных данных, код по ЕГРПОУ, КВЭД, КОАТУУ, код ГНИ, рег. номер уч. карточки руководителя, рег. номер уч. карточки главного бухгалтера |
| 3. | Земельный участок | Код участка, код свидетельства, код категории, НДО ед. площади зем. участка, годовая сумма земельного налога, код налоговой ставки, код льготы, годовая сумма земельного налога (к оплате) |

Выходным документом является налоговая квитанция, в которой содержится информация из вышеупомянутых объектов.

**4.1.4 Алгоритм работы функции**

Расчет годовой суммы налога происходит по формуле:

(НДО + (НДО/100) \* ставка налога) \* площадь (м кв.) \* 12 (4.1)

Величина налога, которая должна быть оплачена получается путем вычитания из годовой суммы льгот.

Пояснения по формуле:

* НДО – нормативно – денежная оценка единицы площади земельного участка;
* Льгота – подразумевается конкретная денежная сумма или же процент от величины налога;
* Налоговая ставка выражается в проценте от НДО;
* 12 – количество месяцев в периоде.

Алгоритм выполнения функции в виде диаграммы деятельности приведен на рисунке 4.1.

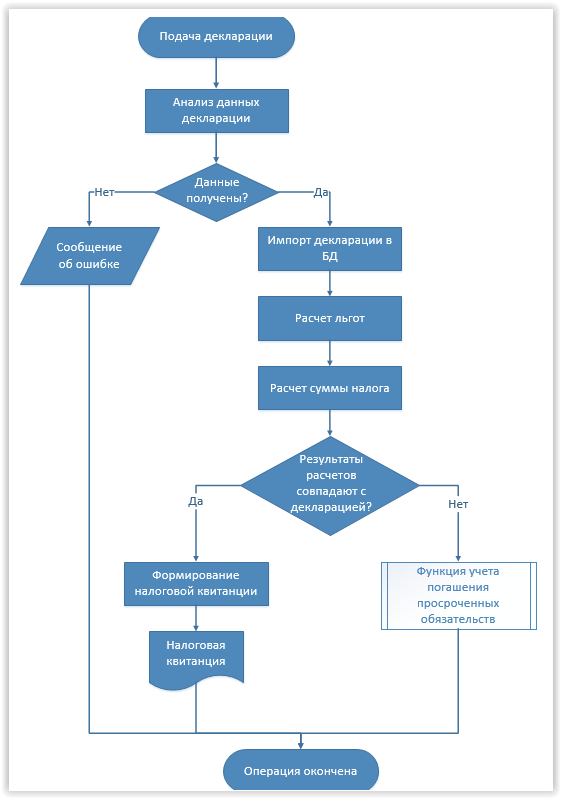


Рисунок 4.1 – Алгоритм функции «Учет налогообложения юридических лиц»

**4.2 Описание алгоритма функции «Учет погашения просроченных налоговых обязательств»**

**4.2.1 Назначения и характеристики функции**

Функция предназначена для реализации алгоритма погашения налоговых обязательств в случае их наличия у налогоплательщика.

Пользователь в лице налогового инспектора руководит процессом путем запуска процедур расчетов, результаты которых будут использоваться при формировании выходных документов: уточненной налоговой декларации и договора об отсрочке денежных обязательств.

Уточненная декларация представляется в электронном формате с расширением «.xml», договор же в свою очередь подается в текстовом формате с расширением «.doc».

Алгоритм применяется в том случае, если у налогоплательщика есть задолженности по выплате налоговых обязательств и на отсрочку их погашения было соответствующее заявление.

**4.2.2 Используемая информация для функции**

Функция в своей работе оперирует данными из базы, перечень используемых объектов приведен в таблице 4.3.

Таблица 4.3 – Используемые таблицы для функции «Учет погашения просроченных налоговых обязательств»

|  |  |  |
| --- | --- | --- |
| № | Наименование таблицы | Поля |
| 1. | Заявление на отсрочку | Код заявления на отсрочку, код юр. лица, тип заявления, наименование долга, сумма отсрочки, срок отсрочки, начало нового периода, конец нового периода, пункт перечня обязательств, раздел перечня обязательств |

Продолжение таблицы 4.3 – Используемые таблицы для функции «Учет погашения просроченных налоговых обязательств»

|  |  |  |
| --- | --- | --- |
| № | Наименование таблицы | Поля |
| 2. | Декларация | Код декларации, начало периода, конец периода |
| 3. | Земельный участок | Код участка, годовая сумма земельного налога (к оплате) |
| 4. | Юридическое лицо | Код лица, наименование (краткое), ФИО руководителя, ФИО бухгалтера, Ид. код по ЕГРПОУ, рег. номер карточки налогоплательщика / серия и номер паспорта руководителя |

**4.2.3 Результаты работы функции «Учет погашения просроченных налоговых обязательств»**

Функция носит еще и вычислительный характер, помимо учетного. Результаты расчетов сохраняются в таблицу «Налоговые обязательства» (поля: сумма штрафа, сумма пени, насчитано до увеличения / уменьшения налогового обязательства, размер увеличения / уменьшения). На основании этих данных происходит формирование выходного документа под названием «Уточненная налоговая декларация».

Также в результате работы функции вносятся данные в таблицу «Решение об отсрочке» (поля: начало / конец установленного периода, установленный срок, проценты, сумма уплаты). Эта информация используется при формулировке договора об отсрочке денежных обязательств.

**4.2.4 Алгоритм работы функции «Учет погашения просроченных налоговых обязательств»**

Расчет пени происходит по формуле:

(недоимка \* кол-во дней просрочки / 100% \* 19% \* 1.2) / 365 (4.2)

Пояснения по формуле:

* недоимка – это размер неуплаты;
* 19% – действующая учетная ставка НБУ;
* 1.2 – это коэффициент для расчёта 120% годовых учётной ставки НБУ, эта цифра не меняется и всегда используется при расчёте;
* 365 – количество календарных дней в году.

Штраф – это величина равная 3% от недоимки.[4]

Алгоритм выполнения функции представлен в виде диаграммы деятельности на рисунке 4.2.

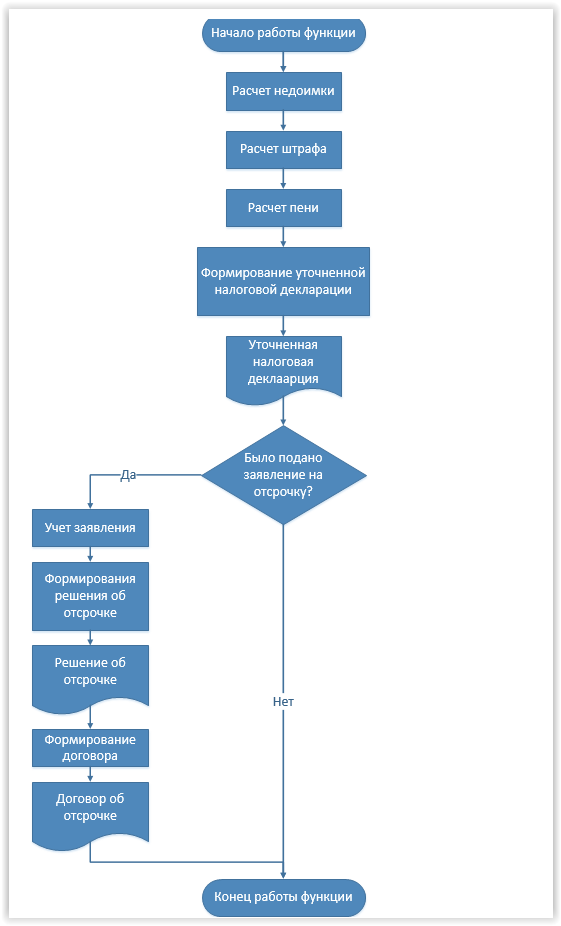


Рисунок 4.2 – Алгоритм функции «Учет погашения просроченных налоговых обязательств»

**5 ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ**

**5.1 Построение объектной модели системы. Выделение основных абстракций системы**

С целью дальнейшей разработки программного обеспечения на основании уже проведенного детального анализа предметной области выделяются абстракции.

Результат в виде перечня абстракций и их описания приведен в таблице 5.1.

Таблица 5.1 – Абстракции информационной системы

|  |  |  |
| --- | --- | --- |
| № | Абстракция | Описание |
| 1. | Декларация | Абстракция, содержащая наименование и код ЕГРПОУ налогоплательщика, ключевые данные о зем. участке, которые необходимы для расчета налога, а также некоторые свойства декларации (год, подписи). |
| 2. | Уточнение декларации | Содержит данные о недоимке, штрафе и пени. |
| 3. | Налоговая квитанция | Содержит наименование и код юр. лица, код и год декларации на основании которой была выдана квитанция и наименование контролирующего органа, который выдал документ. |

Анализ поведения выделенных абстракций даст возможность определить то как они будут себя вести для обеспечения функционала подсистемы. Результата анализа поведения абстракций приведен в таблице 5.2.

Таблица 5.2 – Поведение абстракций информационной системы

|  |  |  |  |
| --- | --- | --- | --- |
| № | Абстракция | Поведение | Описание поведения |
| 1. | Декларация | Запись информации в БД | Данные считываются со входного файла и записываются в базу. |
| 2. | Уточненная декларация | Считывание, обработка и запись информации в БД | На основе записанных данных, а также данных со справочников производится расчет и его результаты так же записывается в базу. |
| 5. | Налоговая квитанция | Считывание данных из БД | При корректном значении налога в исходном файле (декларации) происходит формирование текстового документа с использованием информации из базы. |

Диаграмма с изображенными абстракциями и связями между ними представлена на рисунке 5.1.

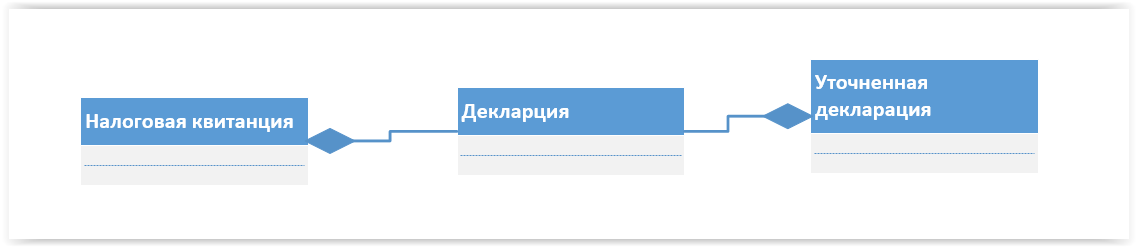


Рисунок 5.1 – Диаграмма абстракций

Установка связей типа «композиция» аргументируется тем, что сущности «Налоговая квитанция и «Уточненная декларация» будут содержать в себе объекты сущности «Декларация», это необходимо для доступа к ее атрибутам.

Для построения объектно – ориентированной модели необходимо выделить классы системы, а также определить их операции и атрибуты.

Описание атрибутов выделенных классов приведено в таблице 5.3.

Таблица 5.3 – Атрибуты классов системы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Атрибут класса | Тип | Обл. видимости | Описание атрибута |
| MainForm.cs | | | | |
| 1. | connectionString | string | private | Строчная переменная с информацией для соединения с сервером базы данных. |
| 2. | conn | OleDb Connection | private | Объект класса, отвечающего за соединение с базой данных. |
| 3. | dcl | Declaration | private | Объект класса «Declaration». |
| 4. | qui | Quittance | private | Объект класса «Quittance». |
| 5. | specDeclForm | Spec\_decl\_ form | private | Объект класса «Spec\_decl\_ form». |
| 6. | reportsForm | ReportForm | private | Объект класса «Report Form». |
| 7. | cur\_area\_row\_ index | int | private | Индекс выбранного земельного участка. |

Продолжение таблицы 5.3 – Атрибуты классов системы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Атрибут класса | Тип | Обл. видимости | Описание атрибута |
| 8. | cur\_company\_ row\_index | int | private | Индекс текущего юридического лица. |
| 9. | cur\_decl\_row\_ index | int | private | Индекс текущей декларации. |
| 10. | decl\_ds | DataSet | private | Набор данных из базы. |
| Declaration.cs | | | | |
| 11. | name | string | private | Наименование юр. лица. |
| 12. | egrpou | string | private | Код по ЕГРПОУ. |
| 13. | year | int | private | Год, за который предоставлена декларация. |
| 14. | sq | float | private | Площадь земельного участка. |
| 15. | ndo | float | private | Нормативно – денежная оценка единицы территории земельного участка. |
| 16. | rate | float | private | Налоговая ставка. |
| 17. | sum | float | private | Сумма налога. |
| 18. | benefit\_sum | float | private | Сумма льгот. |
| 19. | sum\_pay | float | private | Сумма налога (к оплате). |
| 20. | sign1 | string | private | Первая подпись. |
| 21. | sign2 | string | private | Вторая подпись. |
| 22. | tax\_insp | string | private | Контролирующий орган. |

Продолжение таблицы 5.3 – Атрибуты классов системы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Атрибут класса | Тип | Обл. видимости | Описание атрибута |
| Spec\_decl\_form.cs | | | | |
| 23. | sq | float | private | Площадь земельного участка для которого формируется уточненная налоговая декларация. |
| 24. | ndo | float | private | Значение НДО, используемое при пересчете земельного налога |
| 25. | rate | float | private | Налоговая ставка для нового расчета. |
| 26. | benefit\_sum | float | private | Сумма льгот для нового расчета. |
| 27. | sum | float | private | Исходная сумма налога |
| 28. | sum\_pay | float | private | Исходная сумма налога (к оплате). |
| 29. | area\_code | int | private | Код зем. участка, для которого формируется уточненная декларация. |
| 30. | sum\_pay\_ recounted | float | private | Сумма налога к оплате после пересчета с внесенными изменениями. |
| 31. | conn | OleDb  Connection | public | Объект класса, отвечающего за соединение с базой данных. |

Продолжение таблицы 5.3 – Атрибуты классов системы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Атрибут класса | Тип | Обл. видимости | Описание атрибута |
| 32. | specDecl | Specified Declaration | private | Объект класса «Specified Declaration». |
| SpecifiedDeclaration.cs | | | | |
| 33. | count\_prev\_decl | float | private | Насчитано по данным предыдущей декларации. |
| 34. | count\_after | float | private | Насчитано после уточнения. |
| 35. | change\_size | float | private | Размер изменения величины налога. |
| 36. | sur | float | private | Величина штрафа. |
| 37. | fine | float | private | Величина пени. |
| 38. | filePath | string | private | Путь к файлу. |
| 39. | area\_code | int | private | Код участка. |
| 40. | obz\_code | int | private | Код налогового обязательства. |
| Quittance.cs | | | | |
| 41. | company\_ name | string | private | Имя юр. лица, для которого печатается квитанция. |
| 42. | egrpou | string | private | Код по ЕГРПОУ юр. лица. |
| 43. | decl\_code | string | private | Код декларации. |
| 44. | tax\_insp\_name | string | private | Наименование контролирующего органа. |
| 45. | decl\_sign1 | string | private | Первая подпись декларации. |
| 46. | decl\_sign2 | string | private | Вторая подпись декларации. |
| 47. | decl\_year | string | private | Год декларации. |

Продолжение таблицы 5.3 – Атрибуты классов системы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Атрибут класса | Тип | Обл. видимости | Описание атрибута |
| 48. | quittance\_ number | string | private | Номер квитанции. |
| ReportForm.cs | | | | |
| 49. | company Report | Company Report | private | Объект класса «CompanyReport». |
| 50. | conn | OleDbConnection | public | Объект для соединения с базой данных. |
| CompanyReport.cs | | | | |
| 52. | reportDataSet | DataSet | public | Набор данных для формирование отчета. |

Описание операций классов приведено в таблице 5.4.

Таблица 5.4 – Операции классов информационной системы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Атрибут | Тип | Обл. видимости | Описание |
| MainForm.cs | | | | |
| 1. | Main\_form() | - | public | Конструктор класса. Открывает соединение с БД, инициализирует компоненты интерфейса и набор данных для компонентов data grid view. |

Продолжение таблицы 5.4 – Операции классов информационной системы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Атрибут | Тип | Обл. видимости | Описание |
| 2. | importButton\_Click(object sender, EventArgs e) | void | private | Обработчик события нажатия кнопки импорта декларации. |
| 3. | checker() | bool | private | Производит проверку расчета налога. |
| 4. | dataBaseSelect(OleDb Connection conn) | DataSet | public | Делает выборку из базы для отображения в компонентах интерфейса data grid view. |
| 5. | formSpecDecl Button\_Click(object sender, EventArgs e) | void | private | Обработчик события нажатия кнопки формирования уточненной декларации. |
| 6. | Main\_form\_ Activated(object sender, EventArgs e) | void | private | Обработчик события перехода формы в фокус пользователя. Обновляет информацию в data grid view. |
| 7. | makeReport Button\_Click(object sender, EventArgs e) | void | private | Обработчик события нажатия кнопки формирования отчета. |

Продолжение таблицы 5.4 – Операции классов информационной системы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Атрибут | Тип | Обл. видимости | Описание |
| Declaration.cs | | | | |
| 8. | Declaration() | - | public | Конструктор класса. |
| 9. | import(string filePath) | void | public | Импортирует декларацию из файла xlsx. |
| 10. | database Insert(OleDbConnection conn) | void | public | Заносит полученную из файла информацию в базу данных. |
| Spec\_decl\_form.cs | | | | |
| 11. | Spec\_decl\_form() | - | public | Конструктор класса. |
| 12. | formSpecDeclButton\_Click(object sender, EventArgs e) | void | private | Обработчик события нажатия кнопки формирования уточненной налоговой декларации. |
| 13. | getData(int cur\_area\_code, float cur\_sq, float cur\_ndo, float cur\_rate, float cur\_benefit\_sum, float cur\_sum, float cur\_sum\_pay) | void | public | Получает информацию, которая будет использоваться при формировании уточненной налоговой декларации. |

Продолжение таблицы 5.4 – Операции классов информационной системы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Атрибут | Тип | Обл. видимости | Описание |
| 14. | recount() | void | public | Производит перерасчет налога с использованием новых данных. |
| 15. | setCatalogs(OleDb Connection conn) | void | public | Извлекает данные из справочников БД для компонентов combo box, эта информация будет использоваться в перерасчете величины налога. |
| SpecifiedDeclaration.cs | | | | |
| 16. | SpecifiedDeclaration(float cur\_count\_prev\_decl, float cur\_count\_after, string cur\_filePath, int cur\_area\_code) | - | public | Конструктор класса. |
| 17. | export() | void | public | Экспортирует данные в документ xlsx. |
| 18. | sp\_dataBase Insert(OleDbConnection conn) | void | public | Помещает результаты перерасчета в базу данных. |

Продолжение таблицы 5.4 – Операции классов информационной системы

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Атрибут | | Тип | | Обл. видимости | | Описание |
| Quittance.cs | | | | | | | |
| 19. | Quittance(string cur\_ quittance\_number, string cur\_company\_name, string cur\_egrpou, string cur\_decl\_code, string cur\_tax\_insp\_name, string cur\_decl\_sign1, string cur\_decl\_sign2, string cur\_decl\_year) | - | | public | | Конструктор класса. | |
| ReportForm.cs | | | | | | | |
| 20. | ReportForm() | - | | public | | Конструктор класса. | |
| 21. | ReportForm\_Load(object sender, EventArgs e) | void | | private | | Обработчик события загрузки формы. | |
| CompanyReport.cs | | | | | | | |
| 22. | CompanyReport() | - | | public | | Конструктор класса. | |
| 23. | getDataForReport(string companyName, OleDbConnection conn) | Data Set | | public | | Инициализирует набор данных для формирования отчета путем запроса к базе данных. | |

Диаграмма классов системы приведена на рисунке 5.2.

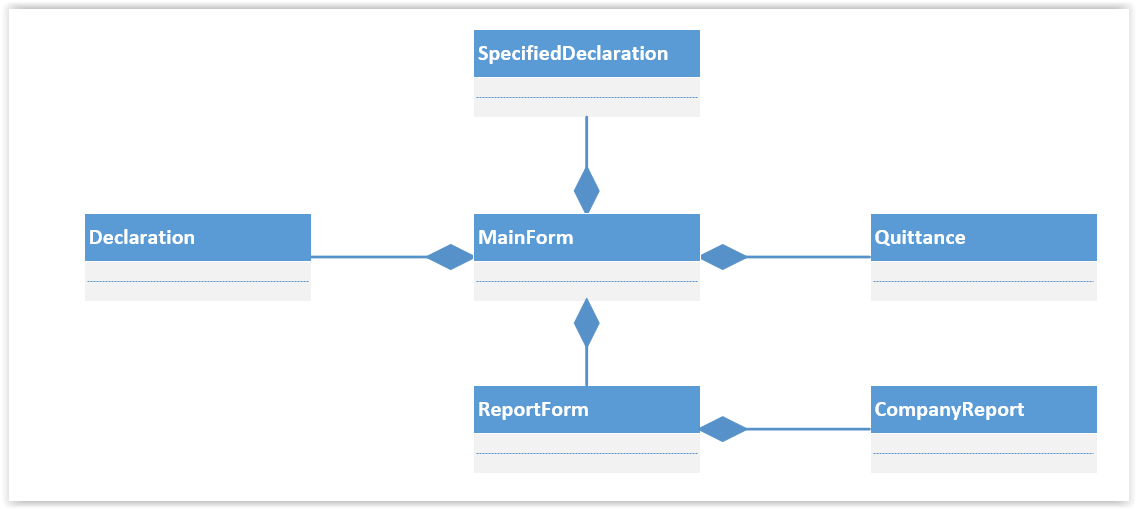


Рисунок 5.2 – Диаграмма классов

**5.2 Разработка интерфейса специального программного обеспечения**

Проектирование и реализация пользовательского интерфейса происходит в среде Microsoft Visual Studio 2013 с использованием технологии Windows Forms.

Интерфейс программы является многодокументным (multiple document interface). Такой тип построения обладает рядом преимуществ:

* В интерфейсе типа MDI общая панель меню и панель инструментов для всех дочерних окон, что уменьшает загромождённость экрана элементами интерфейса и увеличивает его полезную площадь;
* Все окна приложения можно прятать/показывать, сворачивать/разворачивать и проводить с ними другие манипуляции, как с одним окном;
* Дочерние окна можно размещать «черепицей» или «каскадом» в главном окне;
* Увеличение скорости и экономия памяти при работе в одном окне, скорость переключения между дочерними окнами также выше, чем между равноправными в среде операционной системы;
* В некоторых приложениях предусмотрены «горячие сочетания клавиш» для быстрой навигации, в частности, для переключения между окнами. Это ещё более повышает скорость и удобство работы с приложением, так как не задействуются дополнительные ресурсы операционной системы;
* Логичное решение, если окна — части одного проекта.
* Недостатком многодокументного интерфейса можно считать отсутствие возможности вынести содержимое дочерних окон на разные мониторы, а также затрудненность в параллельной работе с несколькими приложениями, в той ситуации, когда необходимо переключаться между внешними окнами одной программы и дочерними другой.

Эскиз главного окна приложения, которое пользователь видит после запуска программы изображен на рисунке 5.3.

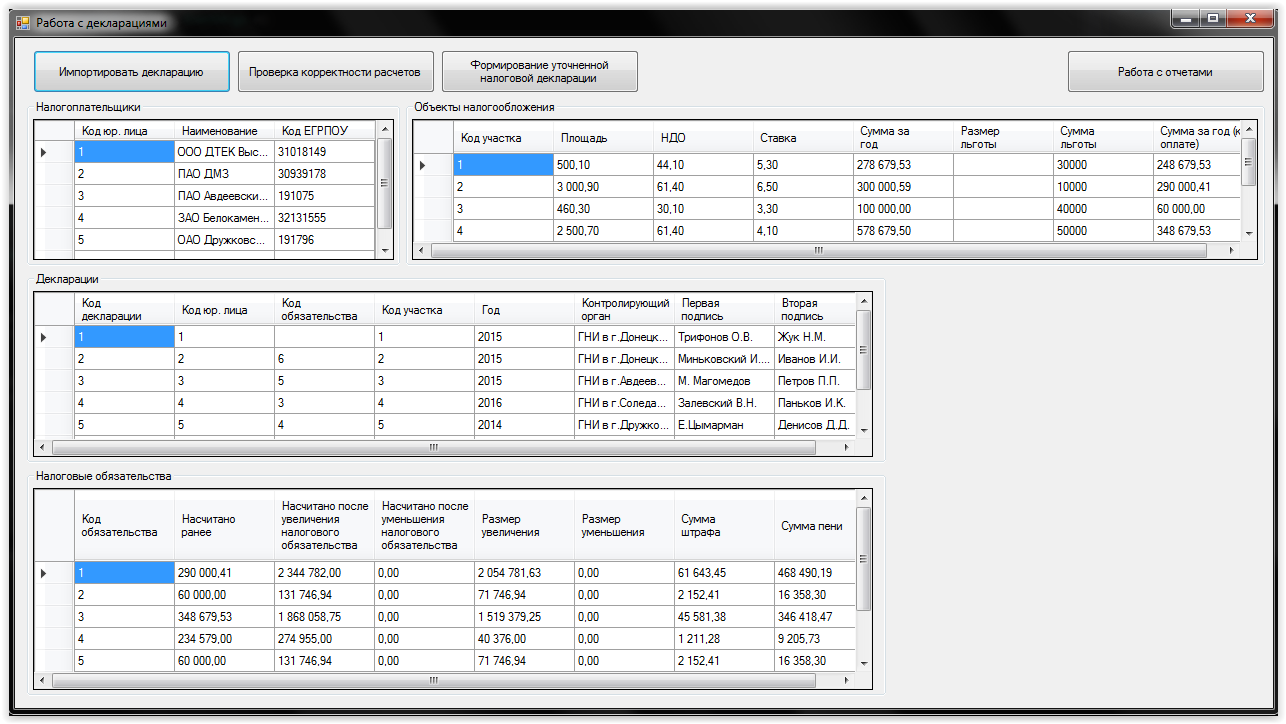


Рисунок 5.3 – Главное окно приложения

При нажатии на кнопку «Импортировать декларацию» пользователь видит модальное окно открытия файла, которое изображено на рисунке 5.4.

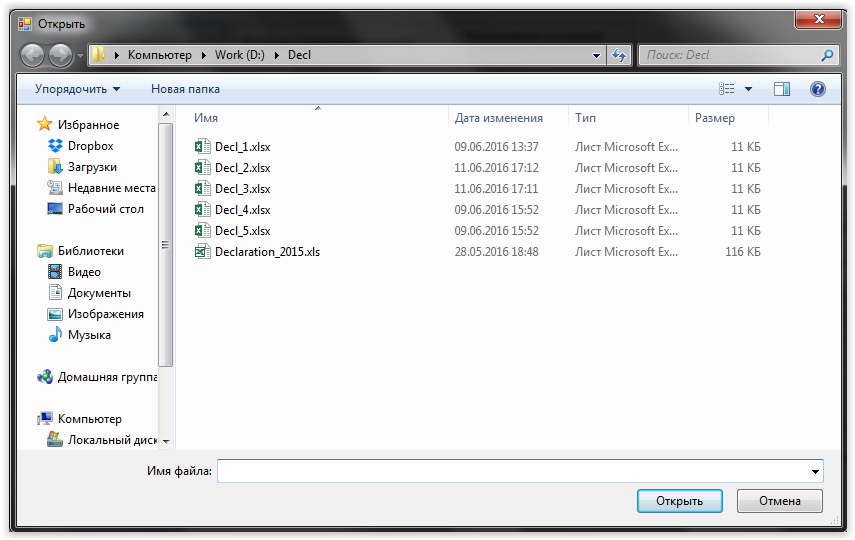


Рисунок 5.4 – Окно открытия файла

При проверке корректности расчетов, в случае положительного результата пользователь видит окно, изображенное на рисунке 5.5, которое предложит сформировать налоговую квитанцию. Если же результат проверки оказался отрицательным, то программа проинформирует об этом, выдав сообщение, показанное на рисунке 5.6.

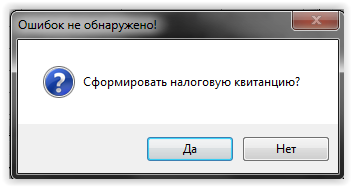


Рисунок 5.5 – Окно, информирующее об отсутствии ошибок

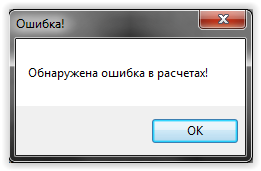


Рисунок 5.6 – Сообщение об ошибке

Формирование уточненной налоговой декларации происходит с использованием данных, которые определяются содержимым элементов combo box, находящихся в окне, представленном на рисунке 5.7.

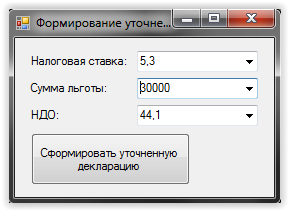


Рисунок 5.7 – Окно формирования уточненной декларации

Отчет создается по нажатию соответствующей кнопки, для его отображения отведена отдельная форма, изображенная на рисунке 5.8.

Форма содержит компонент для представления отчета report viewer.

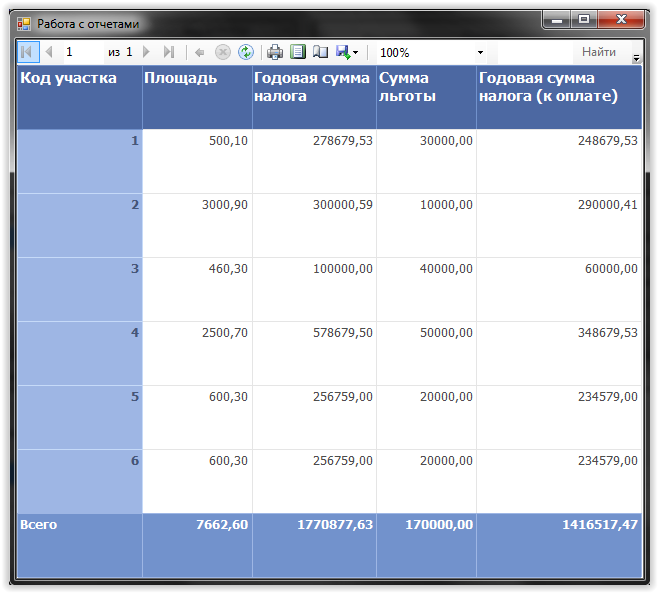


Рисунок 5.8 – Форма для работы с отчетом

**5.3 Описание структуры программы и организации меню**

Программа разрабатывается средствами объектно – ориентированного языка C Sharp, с использованием технологии Windows Forms в среде Microsoft Visual Studio 2013. Для отображения данных в удобном для пользователя формате используются компоненты data grid view.

Соединение и взаимодействие с базой данных обеспечивает технология Object Linking and Embedding, Database (OLE DB). OLE DB является универсальным средством работы с данными. [2]

OLE DB провайдеры могут работать не только с реляционными базами данных, но и с любой другой информацией которую можно представить в табличном виде, например, OLE DB провайдер для служб каталогов или Jet OLE DB провайдер, позволяющий извлекать данные из Excel и Outlook.[5]

По большому счёту, все объекты в OLE DB делятся на 2 группы: Потребители и Поставщики (consumers и providers, соответственно). Суть концепции "Поставщика" в представлении любого необходимого набора данных в виде совокупности строк упорядоченной некоторым образом информации. Концепция "Потребителя" ещё проще: данные, предоставляемые "Поставщиком", используются для извлечения из них информации общепринятым образом: при помощи запросов или манипуляции с отдельным полями структур, которыми физически являются строки, предоставляемые "Поставщиком".

Для работы с файлами Excel (.xlsx) и текстовыми документами (.doc) используются средства библиотеки Microsoft Office Interop.

Разрабатываемое приложение имеет модульную структуру, которая содержит в себе экранные формы, файлы исходного кода классов и отчеты.

Модуль «MainForm.cs» (см. Приложение В) обеспечивает реализацию главной экранной формы. Он содержит в себе методы, обеспечивающие синхронизацию приложения с базой данных, проверку полученных данных, а также обработчики событий нажатия кнопок на экранной форме. Отображение информации обеспечивают компоненты data grid view.

«Declaration.cs» (см. Приложение Д) представляет собой класс, свойства которого соответствуют полям декларации, а методы обеспечивают считывание данных с файла расширения .xlsx и их последующий импорт в базу данных.

Класс «Quittance.cs» (см. Приложение Ж) обеспечивает запись необходимой информации в текстовый файл (квитанцию).

«Spec\_decl\_form.cs» (см. Приложение Г) реализует форму, которая предлагает пользователю выбрать параметры, на основании которых происходит формирование уточненной декларации. Методы класса позволяют получить данные для расчетов, определить источники справочной информации для элементов combo box, произвести перерасчет величины налога, также среди операций есть обработчик события нажатия кнопки формирования уточненной декларации.

Методы класса «SpecifiedDeclaration.cs» (см. Приложение Е) позволяют внести результаты перерасчета налога в базу данных и выбранный файл с расширением .xlsx.

«ReportForm.cs» (см. Приложение И) является классом формы, которая предназначена для отображения отчетов с использованием элемента интерфейса report viewer. При загрузке формы происходит инициализация набора данных для отчета, путем вызова метода класса «CompanyReport.cs» (см. Приложение К), который организует запрос на выборку данных из базы. [1]

Диаграмма иерархии программных модулей представлена на рисунке 5.9.

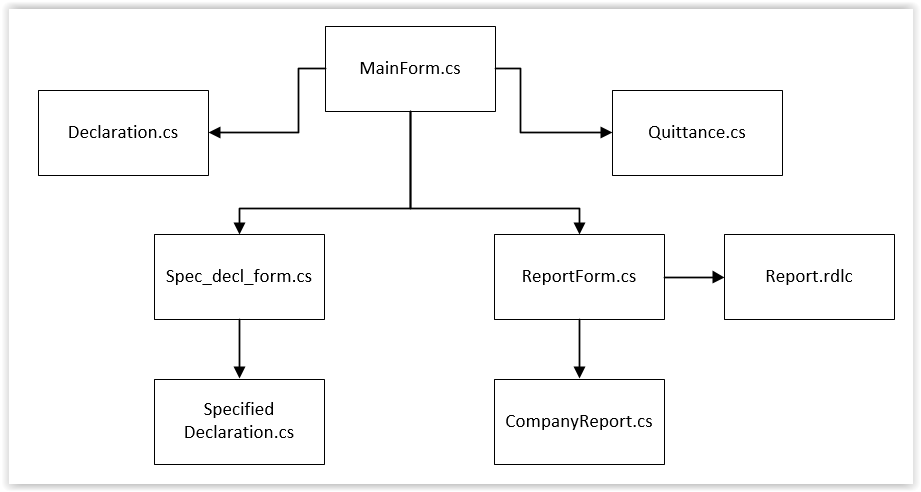


Рисунок 5.9 – Иерархическая структура программы

**5.4 Программный модуль формирования отчетов**

Построение отчета происходит с использованием средств библиотеки Microsoft Reporting WinForms, представление в окне осуществляется при помощи компонента Windows Forms – report viewer.

Выбранная технология позволяет программным способом задать набор данных, который в последствии будет использоваться для построения отчета.

Средство отображения report viewer обладает встроенными функциями, для которых на форму вынесены отдельные кнопки: печать, обновление содержимого, переключение страниц отчета, масштаб, поиск по определенному значению и т. д.

Отчет формируется посредством выполнения запроса к базе данных, на основе которого программным путем создается объект класса Data Set и устанавливается в качестве источника данных для отчета. [6]

Структура файла отчета Report.rdlc показана на рисунке 5.10. Форма для работы с отчетом была приведена ранее на рисунке 5.8.



Рисунок 5.10 – Структура отчета

**5.5 Тестирование программы на контрольном примере**

Тестирование программы происходит на контрольном примере, который представляет собой набор из 5 деклараций, каждая из которых является файлом Microsoft Excel с расширением .xlsx.

Первая декларация корректно заполнена, что подтверждает проверка, выполняемая программой. Вид декларации приведен на рисунке 5.11.

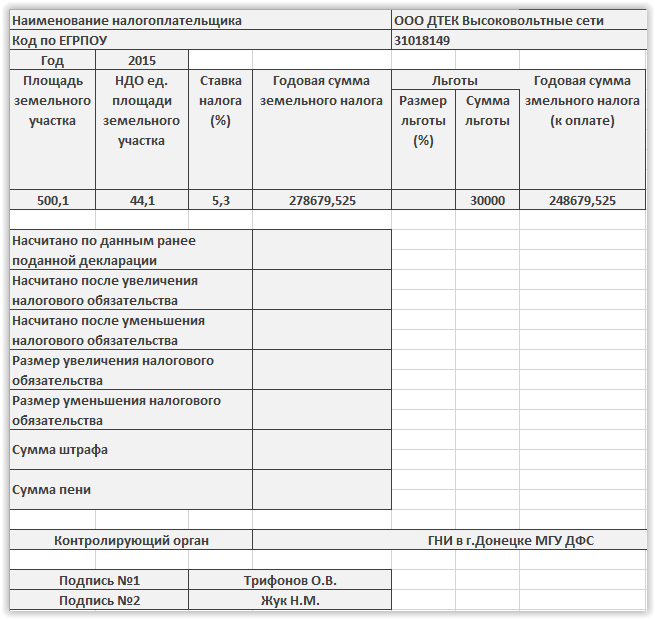


Рисунок 5.11 – Декларация

Результат проверки показан на рисунке 5.12. Также на рисунке присутствуют элементы data grid view, в которых содержатся данные, полученные в результате импорта декларации в базу данных.

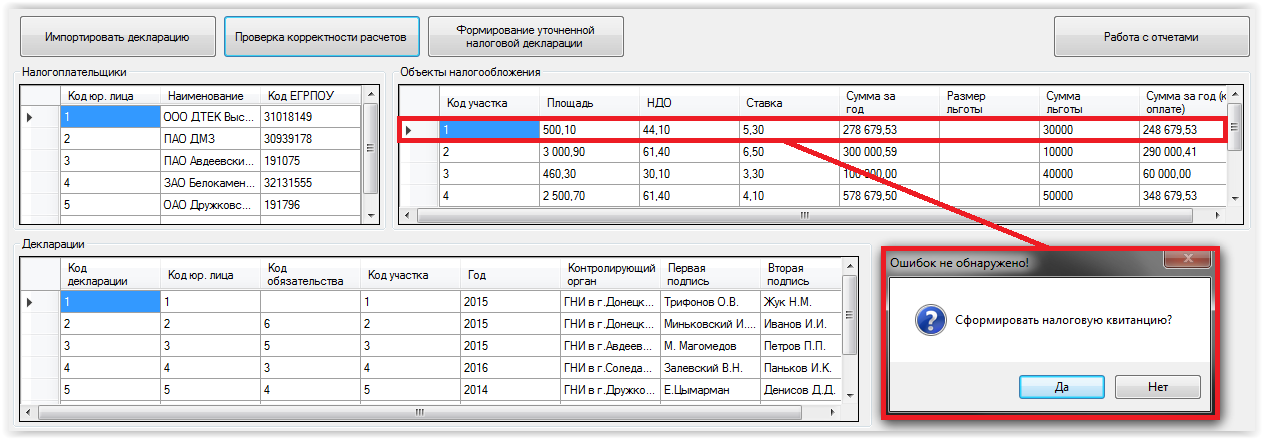


Рисунок 5.12 – Положительный результат проверки

В случае положительного результата проверки пользователю будет предложено сформировать налоговую квитанцию в виде текстового документа с расширением .doc. Содержимое квитанции приведено на рисунке 5.13

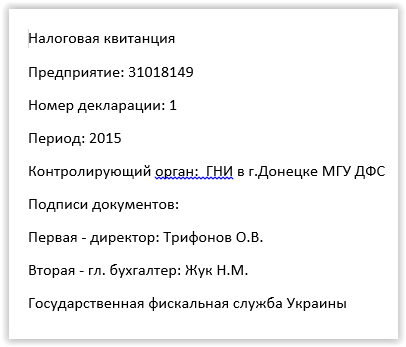


Рисунок 5.13 – Налоговая квитанция

Остальные декларации содержат корректные значения показателей НДО, ставки и льгот, однако конечное значение налога является случайным числом, при проверке фиксируется ошибка, это показано на рисунке 5.14.

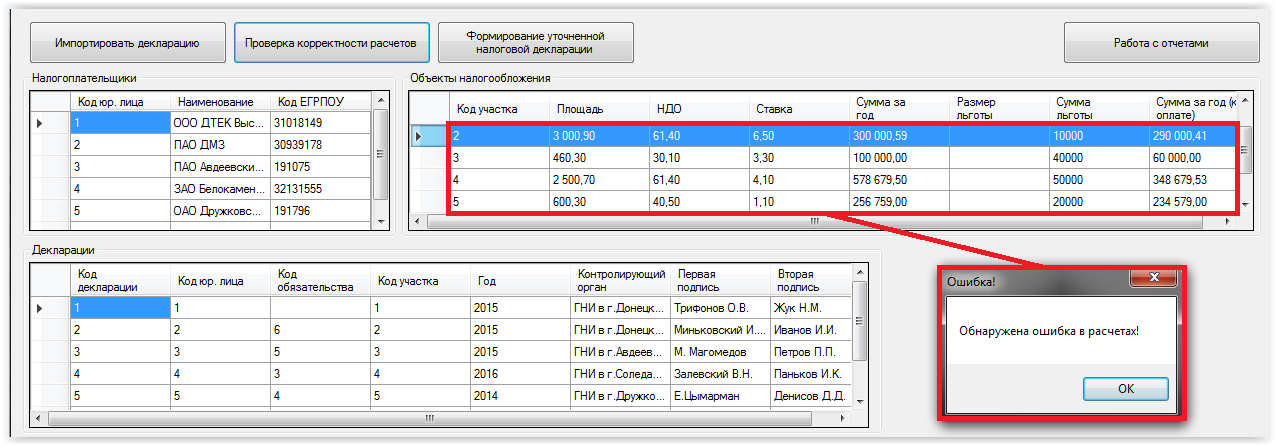


Рисунок 5.14 – Отрицательный результат проверки

Для тестирования функции уточнения деклараций в пятом примере результатом представлено заведомо большее число.

В результате формирования уточненной декларации, данные заносятся в data grid view и имеют вид, представленный на рисунке 5.15.

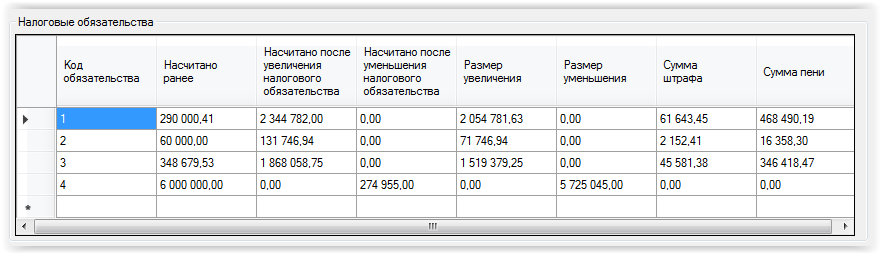


Рисунок 5.15 – Результаты уточнения деклараций

Помимо занесения в базу данных, результаты уточнения также экспортируются в файл Excel, вид файла четвертой декларации после экспорта приведен на рисунке 5.16.

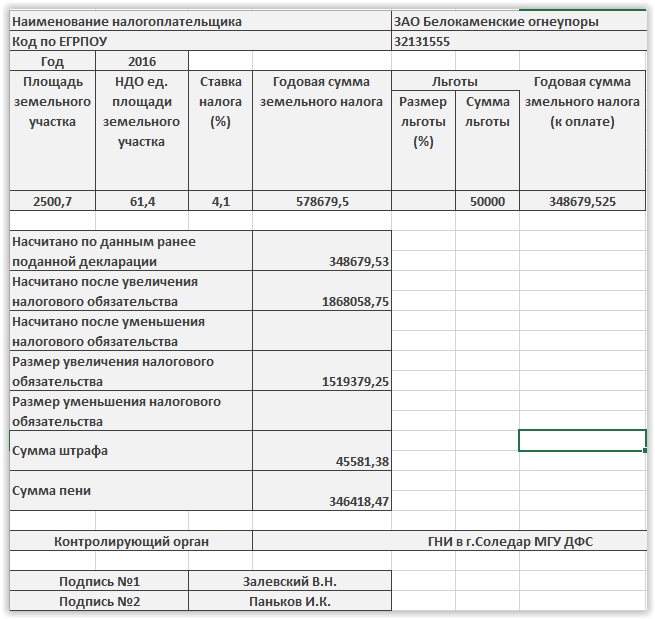


Рисунок 5.16 – Уточненная декларация

В процессе выполнения программы на основе данных, содержащихся в базе формируется отчет, его конечный вид приведен на рисунке 5.17.



Рисунок 5.17 – Отчет

Оценивая результаты тестирования работы программы можно с уверенностью сказать, что она полностью работоспособна: информация из файлов корректно считывается и заносится в базу, результаты расчетов, проводимых программой соответствуют действительности. Выходные документы формируются и корректно отображаются.

**6 ОХРАНА ТРУДА**

**6.1 Анализ условий труда компьютерной аудитории кафедры АСУ**

Темой дипломной работы является разработка компьютеризированной подсистемы начисления и учета платежей земельного налога в условиях районной налоговой инспекции. Выполнение дипломной работы происходит в компьютерной аудитории.

Компьютерная аудитория – это помещение, предназначенное для выполнения работ при помощи компьютерного обеспечения.

К основным видам деятельности компьютерной аудитории относятся: разработка разнообразных IT проектов; выполнение студентами лабораторных работ во время аудиторных занятий; тестирование и экзамены студентов по соответствующим предметам.

Деятельность пользователей компьютерной аудитории напрямую связана с работой на компьютерном оборудовании (ноутбук или стационарный компьютер) и использованием мобильных телефонов. К основным вредным и опасным факторам производственной среды, связанных с работой на персональном компьютере принадлежат: напряжение органов зрения и вызванное им переутомление; значительная нагрузка на кисти рук и пальцы; продолжительное нахождение в сидячей позе, что вызывает дискомфорт в организме; излучения разного вида (рентгеновское, электромагнитное, инфракрасное, статические поля); механические шумы, связанные с работой кулера, дискового привода, оргтехники; ионизация воздуха; выделения в воздух рабочего помещения разных химических веществ.

Мобильный телефон также является источником электромагнитного излучения, на которое организм определенным образом реагирует. Научные опыты подтверждают отрицательное воздействие таких устройств на мозг. Но воздействие мобильного аппарата на организм пользователя и до сих пор не является достаточно изученным.

К психологическим вредным факторам, которые воздействуют на человека при работе с компьютером можно отнести умственное напряжение и нервное перенапряжение, которые возникают в следствии повышенной концентрации внимания.

Все эти факторы негативно влияют на здоровье пользователей компьютерной аудитории и способствуют возникновению профессиональных заболеваний: компьютерный зрительный синдром; радиоволновая болезнь; синдром высыхания роговицы глаза; кистевой туннельный синдром; заболевания кожи; заболевания кишечного тракта; сердечно-сосудистые заболевания.

Негативными последствиями регулярного использования мобильного телефона являются: ослабление памяти; частые головные боли; снижение внимания; напряжение в барабанных перепонках; нарушение сна; раздражительность и др.

**6.2 Меры по улучшению условий труда**

Для уменьшения вредного воздействия негативных факторов производственной среды на пользователей компьютерной аудитории необходимо использовать средства по улучшению их условий труда. Размещение рабочих мест с ПК в подвальных помещениях и на цокольных этажах запрещено, комната, в которой размещено рабочее место с ПК должна иметь природное освещение, которое осуществляется через световые прорези, ориентированные преимущественно на север или северо-восток. Оконные прорези такого помещения должны быть обустроены регулирующимися устройствами (жалюзи, занавески, внешние козырьки). Также помещение должно быть обустроено системой отопления, кондиционирования воздуха, или приливно-вытяжной вентиляцией для обеспечения оптимальных показателей микроклимата. Общий контур заземления строения должен быть выведен через розетки на каждое рабочее место с ПК. Запрещается для улучшения интерьера помещений с ПК использоваться полимерные материалы (деревянно-стружковые плиты, обои, которые моются, рулонные синтетические материалы, бумажный пластик, и т.д.), которые выделяют в воздух вредные химические вещества. В помещениях, где расположены мониторы, нужно выполнять меры по борьбе с статическим полем. Наиболее простым способом соответственно рекомендациям является поддержка относительной влажности воздуха на уровне 50-60%, заземления всех устройств, а также использования пола антистатического линолеума.

Площадь на одно рабочее место с ПК соответственно должно составлять не менее, чем 6,0 м2, а объем не менее, чем 20,0 м3. Расстояние от рабочего места с ПК до стены с окном должна составлять не менее, чем 1,5 м; от других стен - на расстоянии 1 м, а расстояние между столами - 1,5 м

**6.3 Расчет заземления общего контура заземления строения, который должен быть выведен через розетку на каждое рабочее место с ПК**

Для защиты от пробоя напряжения на ПК необходимо сделать расчет защитного заземления. Допустимое заземления для оборудования с напряжением до 1000 В не должно превышать 4 Ом.

Определим расчетное сопротивление почвы по формуле:

=· (6.1)

где  - сопротивление почвы;

 = 100 Ом,  = 1,5.

= 150 Ом.

Определим сопротивление одного заземлителя по формуле:

R0= (6.2)

где  – длина заземлителя  = 3 м;

d = диаметр заземлителя, d = 0,12 м;

Н – расстояние от поверхности до середины заземлителя, Н = 1,5 м.

R0 = 35,5 Ом.

Определим количество параллельно соединенных заземлителей при помощи формулы:

n= R0/  (6.3)

где  = 4 Ом;

 - коэффициент использования группового заземлителя,  = 0, 55.

n = 4,88.

Определим длину горизонтальной соединительной полосы по формуле:

l1=a·(n-1) (6.4)

где а – расстояние между заземлителями, а = 4 – 7 м.

l1 = 15,52 м.

Определим сопротивление соединительной полосы по формуле:

= (/2··l1) ·ln (6.5)

где  – толщина полосы (0,06 м);

hпол – коэффициент использования полосы (0,8).

 = 12,78 Ом.

Определим результирующее сопротивление заземляющего устройства по формуле:

R= (6.6)

R = 7,2 Ом.

Такой показатель превышает допустимую норму, поэтому продолжим расчет, увеличив количество параллельно соединенных заземлителей до 8.

Получим R = 2,5 Ом. Такое значение является вполне приемлемым.

**6.3 Пожарная безопасность**

Самым важным условием работы любого предприятия является соблюдение правил пожарной охраны. В помещении компьютерной аудитории основные меры для обеспечения пожарной безопасности определяет «Инструкция о мерах пожарной безопасности для служебных помещений». Она является обязательной для исполнения всеми пользователями компьютерной аудитории. В инструкции о мерах пожарной безопасности для служебных помещений запрещается:

1. устраивать временные электросети, применять самодельные плавкие вставки в предохранителях, прокладывать электрические провода непосредственно по горючей основе эксплуатировать светильники со снятыми колпаками (рассеивателями), использовать самодельные удлинители, не соответствующие требованиям «Правил устройства электроустановок»;
2. использовать бытовые электрокипятильники, чайники без негорючих подставок, оставлять без присмотра включенными в электросеть кондиционеры, компьютеры, счетные и пишущие машинки и т. п.;
3. загромождать подступы к средствам пожаротушения, использовать пожарные краны, рукава и пожарный инвентарь не по назначению;

Определение видов и количества огнетушителей следует проводить с учетом физико-химических и пожароопасных свойств горючих веществ, площадей и категории производственных помещений по взрывопожарной опасности, а также класса возможного пожара.

В моем случае производственное помещение, где расположены компьютеры, занимает площадь 40 м2. Категория помещения по взрывопожарной и пожарной опасности - В, поскольку в нем находятся твердые горючие материалы (бумага, мебель и т.д.). Класс возможного пожара - А. Это помещение, с учетом минимальной порчи компьютерной техники при тушении пожара желательно оснастить двумя углекислотными огнетушителями типа ОУ - 5.

**ЗАКЛЮЧЕНИЕ**

Результатом выполнения выпускной квалификационной работы является проект автоматизированной системы учета налога на землю в рамках районной налоговой инспекции. Выполнение работы состояло из нескольких этапов, которые позволили провести детальный анализ предметной области, определить функции системы, выделить сущности и на их основе построить модель данных.

Конечный результат выполнения работы являет собой полностью разработанную базу данных, а также программу, реализующую основные функции системы. Программа и база данных полностью работоспособны, что подтверждает тестирование на контрольном примере.

Результаты полученные в ходе разработки проекта могут стать базой для дальнейшей работы в этой предметной области, которая позволит расширить функционал системы, сделать ее еще более разветвленной и, возможно, значительно расширить спектр ее применения.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

* 1. Сообщество программистов [Электронный ресурс] // StackOverFlow.com: Stack Exchange Inc. Режим доступа: http://stackoverflow.com/questions/2263566/how-do-i-programmatically-set-data-source-for-asp-net-reportviewer-control (дата обращения 10.06.2016);
  2. Форум программистов [Электронный ресурс] // CyberForum.ru: vBulletin Solutions, Inc. Режим доступа: http://www.cyberforum.ru/ado-net/thread182279.html#post1061537 (дата обращения 31.05.2016);
  3. Электронная энциклопедия [Электронный ресурс] // Wikipedia.org: Wikimedia Foundation, Inc. Режим доступа: https://ru.wikipedia.org/wiki/Клиент-сервер   
     (дата обращения 16.05.2016);
  4. Электронный справочник для бухгалтеров Smart [Электронный ресурс] // smcs.com.ua: Cleverfox Ltd. Режим доступа:   
     https://help.smcs.com.ua/ru/reader/articles/read/12213/kak-rasschitat-penyu  
     (дата обращения 21.05.2016);
  5. База знаний [Электронный ресурс] // ConnectionStrings.com: ConnectionStrings.com. Режим доступа: http://www.connectionstrings.com/microsoft-ole-db-provider-for-sql-server-sqloledb

(дата обращения 07.06.2016);

* 1. Сообщество программистов [Электронный ресурс] // codeproject.com: CodeProject. Режим доступа: http://www.codeproject.com/Questions/245159/progamatically-giving-data-source-to-microsoft-rep

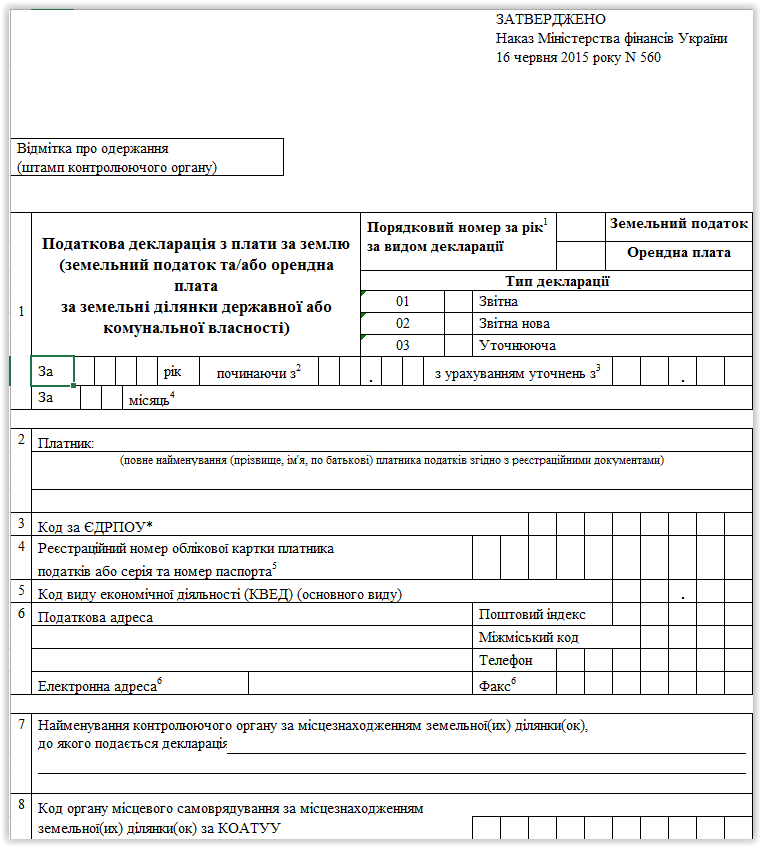
**ПРИЛОЖЕНИЕ А  
Налоговая декларация**

Рисунок А.1 – Первая страница налоговой декларации

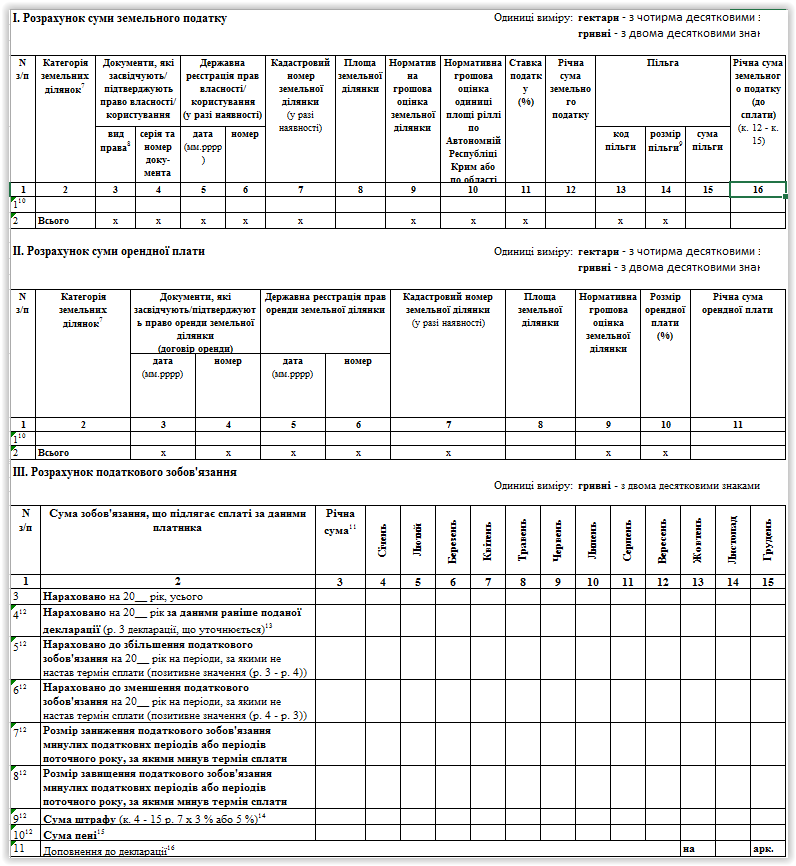


Рисунок А.2 – Вторая страница налоговой декларации

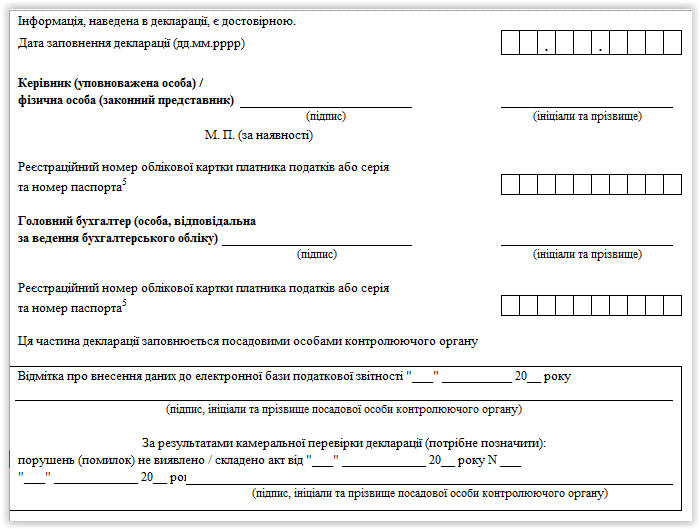


Рисунок А.3 – Третья страница налоговой декларации

**ПРИЛОЖЕНИЕ Б**

**Правоустанавливающие документы на землю**



Рисунок Б.1 – Правоустанавливающие документы на землю

**ПРИЛОЖЕНИЕ В**

**Листинг программного модуля «MainForm.cs»**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

using System.IO;

using Microsoft.Office.Interop.Excel;

namespace Diplom

{

public partial class Main\_form : Form

{

private string connectionString = @"Provider=SQLOLEDB; Data Source=ULTHWE-PC\SQLEXPRESS; Initial Catalog = Diplom\_demo; Integrated Security=SSPI; Trusted Connection = Yes;";

private OleDbConnection conn;

private Declaration dcl;

private Quittance qui;

private Spec\_decl\_form specDeclForm;

private ReportForm reportsForm;

private int cur\_area\_row\_index;

private int cur\_company\_row\_index;

private int cur\_decl\_row\_index;

private DataSet decl\_ds;

public Main\_form()

{

InitializeComponent();

conn = new OleDbConnection(connectionString);

conn.Open();

decl\_ds = dataBaseSelect(conn);

DataGridViewCellStyle style = new DataGridViewCellStyle();

style.Format = "N2";

dgv\_company.DataSource = decl\_ds.Tables["Yur\_litso"];

dgv\_area.DataSource = decl\_ds.Tables["Zem\_uch"];

dgv\_decl.DataSource = decl\_ds.Tables["Dekl"];

dgv\_obz.DataSource = decl\_ds.Tables["Nalog\_obz"];

dgv\_company.Columns[0].HeaderCell.Value = "Код юр. лица";

dgv\_company.Columns[1].HeaderCell.Value = "Наименование";

dgv\_company.Columns[2].HeaderCell.Value = "Код ЕГРПОУ";

dgv\_area.Columns[0].HeaderCell.Value = "Код участка";

dgv\_area.Columns[1].HeaderCell.Value = "Площадь";

dgv\_area.Columns[1].DefaultCellStyle = style;

dgv\_area.Columns[2].HeaderCell.Value = "НДО";

dgv\_area.Columns[2].DefaultCellStyle = style;

dgv\_area.Columns[3].HeaderCell.Value = "Ставка";

dgv\_area.Columns[3].DefaultCellStyle = style;

dgv\_area.Columns[4].HeaderCell.Value = "Сумма за год";

dgv\_area.Columns[4].DefaultCellStyle = style;

dgv\_area.Columns[5].HeaderCell.Value = "Размер льготы";

dgv\_area.Columns[6].HeaderCell.Value = "Сумма льготы";

dgv\_area.Columns[7].HeaderCell.Value = "Сумма за год (к оплате)";

dgv\_area.Columns[7].DefaultCellStyle = style;

dgv\_decl.Columns[0].HeaderCell.Value = "Код декларации";

dgv\_decl.Columns[1].HeaderCell.Value = "Код юр. лица";

dgv\_decl.Columns[2].HeaderCell.Value = "Код обязательства";

dgv\_decl.Columns[3].HeaderCell.Value = "Код участка";

dgv\_decl.Columns[4].HeaderCell.Value = "Год";

dgv\_decl.Columns[5].HeaderCell.Value = "Контролирующий орган";

dgv\_decl.Columns[6].HeaderCell.Value = "Первая подпись";

dgv\_decl.Columns[7].HeaderCell.Value = "Вторая подпись";

dgv\_obz.Columns[0].HeaderCell.Value = "Код обязательства";

dgv\_obz.Columns[1].HeaderCell.Value = "Насчитано ранее";

dgv\_obz.Columns[1].DefaultCellStyle = style;

dgv\_obz.Columns[2].HeaderCell.Value = "Насчитано после увеличения налогового обязательства";

dgv\_obz.Columns[2].DefaultCellStyle = style;

dgv\_obz.Columns[3].HeaderCell.Value = "Насчитано после уменьшения налогового обязательства";

dgv\_obz.Columns[3].DefaultCellStyle = style;

dgv\_obz.Columns[4].HeaderCell.Value = "Размер увеличения";

dgv\_obz.Columns[4].DefaultCellStyle = style;

dgv\_obz.Columns[5].HeaderCell.Value = "Размер уменьшения";

dgv\_obz.Columns[5].DefaultCellStyle = style;

dgv\_obz.Columns[6].HeaderCell.Value = "Сумма штрафа";

dgv\_obz.Columns[6].DefaultCellStyle = style;

dgv\_obz.Columns[7].HeaderCell.Value = "Сумма пени";

dgv\_obz.Columns[7].DefaultCellStyle = style;

}

private void Main\_form\_Load(object sender, EventArgs e)

{

}

private void importButton\_Click(object sender, EventArgs e)

{

dcl = new Declaration();

OpenFileDialog openDialog = new OpenFileDialog();

if (openDialog.ShowDialog() == DialogResult.OK)

{

string filePath = openDialog.FileName;

dcl.import(filePath);

dcl.dataBaseInsert(conn);

decl\_ds = dataBaseSelect(conn);

dgv\_company.DataSource = decl\_ds.Tables["Yur\_litso"];

dgv\_area.DataSource = decl\_ds.Tables["Zem\_uch"];

dgv\_decl.DataSource = decl\_ds.Tables["Dekl"];

dgv\_obz.DataSource = decl\_ds.Tables["Nalog\_obz"];

}

}

private void checkButton\_Click(object sender, EventArgs e)

{

if (!checker())

{

MessageBox.Show("Обнаружена ошибка в расчетах!", "Ошибка!");

}

else if (MessageBox.Show("Сформировать налоговую квитанцию?", "Ошибок не обнаружено!", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

for (int i = 0; i < dgv\_decl.RowCount-1; i++)

if (dgv\_decl.Rows[i].Cells[3].Value.ToString().Equals(dgv\_area.Rows[cur\_area\_row\_index].Cells[0].Value.ToString()))

cur\_decl\_row\_index = i;

for (int i = 0; i < dgv\_company.RowCount-1; i++)

if (dgv\_decl.Rows[cur\_decl\_row\_index].Cells[1].Value.ToString().Equals(dgv\_company.Rows[i].Cells[0].Value.ToString()))

cur\_company\_row\_index = i;

string cur\_companu\_name = dgv\_company.Rows[cur\_company\_row\_index].Cells[1].Value.ToString();

string cur\_egrpou = dgv\_company.Rows[cur\_company\_row\_index].Cells[2].Value.ToString();

string cur\_decl\_code = dgv\_decl.Rows[cur\_decl\_row\_index].Cells[0].Value.ToString();

string cur\_tax\_insp\_name = dgv\_decl.Rows[cur\_decl\_row\_index].Cells[5].Value.ToString();

string cur\_decl\_sign1 = dgv\_decl.Rows[cur\_decl\_row\_index].Cells[6].Value.ToString();

string cur\_decl\_sign2 = dgv\_decl.Rows[cur\_decl\_row\_index].Cells[7].Value.ToString();

string cur\_decl\_year = dgv\_decl.Rows[cur\_decl\_row\_index].Cells[4].Value.ToString();

string cur\_quittance\_number = "" + dgv\_area.Rows[cur\_area\_row\_index].Cells[0].Value +

dgv\_decl.Rows[cur\_decl\_row\_index].Cells[1].Value +

dgv\_decl.Rows[cur\_decl\_row\_index].Cells[3].Value;

qui = new Quittance(cur\_quittance\_number, cur\_companu\_name, cur\_egrpou, cur\_decl\_code,

cur\_tax\_insp\_name, cur\_decl\_sign1, cur\_decl\_sign2, cur\_decl\_year);

}

}

private bool checker()

{

int result;

cur\_area\_row\_index = dgv\_area.CurrentRow.Index;

result = (int)(((Convert.ToDouble(dgv\_area.Rows[cur\_area\_row\_index].Cells[2].Value) +

(Convert.ToDouble(dgv\_area.Rows[cur\_area\_row\_index].Cells[2].Value) / 100 \*

Convert.ToDouble(dgv\_area.Rows[cur\_area\_row\_index].Cells[3].Value))) \*

Convert.ToDouble(dgv\_area.Rows[cur\_area\_row\_index].Cells[1].Value) \* 12) -

Convert.ToDouble(dgv\_area.Rows[cur\_area\_row\_index].Cells[6].Value));

if (result == (int)Convert.ToDouble(dgv\_area.Rows[cur\_area\_row\_index].Cells[7].Value))

return true;

else return false;

}

public DataSet dataBaseSelect(OleDbConnection conn)

{

string select\_command\_company = "SELECT \* FROM dbo.Yur\_litso";

string select\_command\_area = "SELECT \* FROM dbo.Zem\_uch";

string select\_command\_declaration = "SELECT \* FROM dbo.Dekl";

string select\_command\_obz = "SELECT \* FROM dbo.Nalog\_obz";

OleDbCommand command\_company = new OleDbCommand(select\_command\_company, conn);

OleDbCommand command\_area = new OleDbCommand(select\_command\_area, conn);

OleDbCommand command\_declaration = new OleDbCommand(select\_command\_declaration, conn);

OleDbCommand command\_obz = new OleDbCommand(select\_command\_obz, conn);

OleDbDataAdapter dataAdapter\_company = new OleDbDataAdapter(command\_company);

OleDbDataAdapter dataAdapter\_area = new OleDbDataAdapter(command\_area);

OleDbDataAdapter dataAdapter\_declaration = new OleDbDataAdapter(command\_declaration);

OleDbDataAdapter dataAdapter\_obz = new OleDbDataAdapter(command\_obz);

DataSet ds = new DataSet();

dataAdapter\_company.TableMappings.Add("Table", "Yur\_litso");

dataAdapter\_area.TableMappings.Add("Table", "Zem\_uch");

dataAdapter\_declaration.TableMappings.Add("Table", "Dekl");

dataAdapter\_obz.TableMappings.Add("Table", "Nalog\_obz");

dataAdapter\_company.Fill(ds);

dataAdapter\_area.Fill(ds);

dataAdapter\_declaration.Fill(ds);

dataAdapter\_obz.Fill(ds);

return ds;

}

private void formSpecDecBlutton\_Click(object sender, EventArgs e)

{

specDeclForm = new Spec\_decl\_form();

specDeclForm.setCatalogs(conn);

specDeclForm.Show();

specDeclForm.conn = conn;

cur\_area\_row\_index = dgv\_area.CurrentRow.Index;

specDeclForm.getData(Convert.ToInt32(dgv\_area.Rows[cur\_area\_row\_index].Cells[0].Value),

(float)Convert.ToDouble(dgv\_area.Rows[cur\_area\_row\_index].Cells[1].Value),

(float)Convert.ToDouble(dgv\_area.Rows[cur\_area\_row\_index].Cells[2].Value),

(float)Convert.ToDouble(dgv\_area.Rows[cur\_area\_row\_index].Cells[3].Value),

(float)Convert.ToDouble(dgv\_area.Rows[cur\_area\_row\_index].Cells[6].Value),

(float)Convert.ToDouble(dgv\_area.Rows[cur\_area\_row\_index].Cells[4].Value),

(float)Convert.ToDouble(dgv\_area.Rows[cur\_area\_row\_index].Cells[7].Value));

}

private void Main\_form\_Activated(object sender, EventArgs e)

{

decl\_ds = dataBaseSelect(conn);

dgv\_company.DataSource = decl\_ds.Tables["Yur\_litso"];

dgv\_area.DataSource = decl\_ds.Tables["Zem\_uch"];

dgv\_decl.DataSource = decl\_ds.Tables["Dekl"];

dgv\_obz.DataSource = decl\_ds.Tables["Nalog\_obz"];

}

private void makeReportButton\_Click(object sender, EventArgs e)

{

reportsForm = new ReportForm();

reportsForm.conn = conn;

reportsForm.Show();

}

}

}

**ПРИЛОЖЕНИЕ Г**

**Листинг программного модуля «Spec\_decl\_form.cs»**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Data;

using System.Threading.Tasks;

using System.Data.OleDb;

using System.IO;

using Microsoft.Office.Interop.Excel;

using System.Windows.Forms;

namespace Diplom

{

class SpecifiedDeclaration

{

private float count\_prev\_decl;// { get; set; }

private float count\_after;// { get; set; }

private float change\_size;// { get; set; }

private float sur;// { get; set; }//штраф

private float fine;// { get; set; }//пеня

private string filePath;

private int area\_code;

private int obz\_code;

public SpecifiedDeclaration(float cur\_count\_prev\_decl, float cur\_count\_after, string cur\_filePath, int cur\_area\_code)

{

this.area\_code = cur\_area\_code;

this.count\_prev\_decl = cur\_count\_prev\_decl;

this.count\_after = cur\_count\_after;

this.change\_size = count\_after - count\_prev\_decl;

this.sur = (change\_size / 100) \* 3;

this.fine = (float)((change\_size / 100) \* 19 \* 1.2);

this.filePath = cur\_filePath;

export();

}

public void export()

{

Microsoft.Office.Interop.Excel.Application ObjExcel = new Microsoft.Office.Interop.Excel.Application();

Workbook ObjWorkBook = ObjExcel.Workbooks.Open(filePath, 0, false, 5, "", "", false, XlPlatform.xlWindows, "", true, false, 0, true, false, false);

Worksheet ObjWorkSheet = (Worksheet)ObjWorkBook.Sheets[1];

Range cell = ObjWorkSheet.get\_Range("D12");

cell.Value = count\_prev\_decl;

if(count\_after > count\_prev\_decl)

{

cell = ObjWorkSheet.get\_Range("D14");

cell.Value = count\_after;

cell = ObjWorkSheet.get\_Range("D18");

cell.Value = change\_size;

cell = ObjWorkSheet.get\_Range("D22");

cell.Value = sur;

cell = ObjWorkSheet.get\_Range("D24");

cell.Value = fine;

}

else

{

cell = ObjWorkSheet.get\_Range("D16");

cell.Value = count\_after;

cell = ObjWorkSheet.get\_Range("D20");

cell.Value = change\_size \* -1;

cell = ObjWorkSheet.get\_Range("D22");

cell.Value = 0;

cell = ObjWorkSheet.get\_Range("D24");

cell.Value = 0;

}

ObjExcel.Quit();

}

public void sp\_dataBaseInsert(OleDbConnection conn)

{

string insert\_command\_str = "INSERT INTO dbo.Nalog\_obz(Count\_pred, Count\_do\_uvel, Count\_do\_umen, Razm\_uvel, Razm\_umen, Shtraf\_sum, Penya\_sum) VALUES (?,?,?,?,?,?,?)";

string update\_command\_str = "UPDATE dbo.Dekl SET Kod\_obz = ? WHERE Kod\_uch = ?";

OleDbCommand insert\_command = new OleDbCommand(insert\_command\_str, conn);

OleDbCommand update\_command = new OleDbCommand(update\_command\_str, conn);

insert\_command.Parameters.AddWithValue("Count\_pred", count\_prev\_decl);

if (count\_after > count\_prev\_decl)

{

insert\_command.Parameters.AddWithValue("Count\_do\_uvel", count\_after);

insert\_command.Parameters.AddWithValue("Count\_do\_umen", 0);

insert\_command.Parameters.AddWithValue("Razm\_uvel", change\_size);

insert\_command.Parameters.AddWithValue("Razm\_umen", 0);

insert\_command.Parameters.AddWithValue("Shtraf\_sum", sur);

insert\_command.Parameters.AddWithValue("Penya\_sum", fine);

}

else

{

insert\_command.Parameters.AddWithValue("Count\_do\_uvel", 0);

insert\_command.Parameters.AddWithValue("Count\_do\_umen", count\_after);

insert\_command.Parameters.AddWithValue("Razm\_uvel", 0);

insert\_command.Parameters.AddWithValue("Razm\_umen", change\_size\*-1);

insert\_command.Parameters.AddWithValue("Shtraf\_sum", 0);

insert\_command.Parameters.AddWithValue("Penya\_sum", 0);

}

insert\_command.ExecuteNonQuery();

string select\_command\_str = "SELECT Kod\_obz FROM dbo.Nalog\_obz";

OleDbCommand select\_command = new OleDbCommand(select\_command\_str, conn);

select\_command.ExecuteNonQuery();

OleDbDataAdapter da = new OleDbDataAdapter(select\_command);

DataSet ds = new DataSet();

da.Fill(ds);

obz\_code = Convert.ToInt32(ds.Tables[0].Rows[ds.Tables[0].Rows.Count-1].ItemArray[0].ToString());

update\_command.Parameters.Add(new OleDbParameter("@obz\_code", obz\_code));

update\_command.Parameters.Add(new OleDbParameter("@current\_area", area\_code));

update\_command.ExecuteNonQuery();

}

}

}

**ПРИЛОЖЕНИЕ Д**

**Листинг программного модуля «Declaration.cs»**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Data;

using System.Threading.Tasks;

using System.Data.OleDb;

using System.IO;

using Microsoft.Office.Interop.Excel;

using System.Windows.Forms;

namespace Diplom

{

class Declaration

{

private string name;// { get; set; }

private string egrpou;// { get; set; }

private int year;// { get; set; }

private float sq;// { get; set; }

private float NDO;// { get; set; }

private float rate;// { get; set; }

private float sum;// { get; set; }

private float benefit\_size;// { get; set; }

private float benefit\_sum;// { get; set; }

private float sum\_pay;//{ get; set; }

private string sign1;// { get; set; }

private string sign2;// { get; set; }

private string tax\_insp;// { get; set; }

//private int counter = 1;

public Declaration()

{

}

public void import(string filePath)

{

Microsoft.Office.Interop.Excel.Application ObjExcel = new Microsoft.Office.Interop.Excel.Application();

Workbook ObjWorkBook = ObjExcel.Workbooks.Open(filePath, 0, false, 5, "", "", false, XlPlatform.xlWindows, "", true, false, 0, true, false, false);

Worksheet ObjWorkSheet = (Worksheet)ObjWorkBook.Sheets[1];

Range cell = ObjWorkSheet.get\_Range("E1");

this.name = cell.Text;

cell = ObjWorkSheet.get\_Range("E2");

this.egrpou = cell.Text;

cell = ObjWorkSheet.get\_Range("B3");

this.year = Convert.ToInt32(cell.Text);

cell = ObjWorkSheet.get\_Range("A10");

this.sq = (float)Convert.ToDouble(cell.Text);

cell = ObjWorkSheet.get\_Range("B10");

this.NDO = (float)Convert.ToDouble(cell.Text);

cell = ObjWorkSheet.get\_Range("C10");

this.rate = (float)Convert.ToDouble(cell.Text);

cell = ObjWorkSheet.get\_Range("D10");

this.sum = (float)Convert.ToDouble(cell.Text);

cell = ObjWorkSheet.get\_Range("E10");

if (cell == null)

{

this.benefit\_size = (float)Convert.ToDouble(cell.Text);

}

else

{

cell = ObjWorkSheet.get\_Range("F10");

this.benefit\_sum = (float)Convert.ToDouble(cell.Text);

}

cell = ObjWorkSheet.get\_Range("G10");

this.sum\_pay = (float)Convert.ToDouble(cell.Text);

cell = ObjWorkSheet.get\_Range("D27");

this.tax\_insp = cell.Text;

cell = ObjWorkSheet.get\_Range("C29");

this.sign1 = cell.Text;

cell = ObjWorkSheet.get\_Range("C30");

this.sign2 = cell.Text;

ObjExcel.Quit();

}

public void dataBaseInsert(OleDbConnection conn)

{

string insert\_command\_company = "INSERT INTO dbo.Yur\_litso(Name, Egrpou) VALUES (?,?)";

string insert\_command\_area = "INSERT INTO dbo.Zem\_uch(Ploshad, Ndo, Stavka, God\_sum, Sum\_lgoti, God\_sum\_pay) VALUES (?,?,?,?,?,?)";

string insert\_command\_declaration = "INSERT INTO dbo.Dekl(Kod\_litsa, Kod\_uch, God, Kontr\_organ, Podp1, Podp2) VALUES (?, ?, ?, ?, ?, ?)";

string select\_name\_command\_company = "SELECT Name FROM dbo.Yur\_litso";

OleDbCommand command\_company = new OleDbCommand(insert\_command\_company, conn);

OleDbCommand command\_area = new OleDbCommand(insert\_command\_area, conn);

OleDbCommand command\_declaration = new OleDbCommand(insert\_command\_declaration, conn);

OleDbCommand command\_company\_select\_name = new OleDbCommand(select\_name\_command\_company, conn);

OleDbDataAdapter da1 = new OleDbDataAdapter(command\_company\_select\_name);

DataSet ds1 = new DataSet();

da1.Fill(ds1);

bool flag = false;

if (ds1.Tables[0].Rows.Count > 0)

{

for (int i = 0; i < ds1.Tables[0].Rows.Count; i++)

if (ds1.Tables[0].Rows[i].ItemArray[0].ToString().Trim().Equals(name))

flag = true;

}

if(!flag)

{

command\_company.Parameters.AddWithValue("Name", name);

command\_company.Parameters.AddWithValue("Egrpou", egrpou);

command\_company.ExecuteNonQuery();

}

command\_area.Parameters.AddWithValue("Ploshad", sq);

command\_area.Parameters.AddWithValue("Ndo", NDO);

command\_area.Parameters.AddWithValue("Stavka", rate);

command\_area.Parameters.AddWithValue("God\_sum", sum);

command\_area.Parameters.AddWithValue("Sum\_lgoti", benefit\_sum);

command\_area.Parameters.AddWithValue("God\_sum\_pay", sum\_pay);

command\_area.ExecuteNonQuery();

string select\_code\_command\_company = "SELECT Kod\_litsa, Name FROM dbo.Yur\_litso";

OleDbCommand command\_company\_select\_code = new OleDbCommand(select\_code\_command\_company, conn);

OleDbDataAdapter da2 = new OleDbDataAdapter(command\_company\_select\_code);

DataSet ds2 = new DataSet();

da2.Fill(ds2);

int index = 0;

for (int i = 0; i < ds2.Tables[0].Rows.Count; i++)

if (ds2.Tables[0].Rows[i].ItemArray[1].ToString().Trim().Equals(name))

index = i;

command\_declaration.Parameters.AddWithValue("Kod\_litsa", ds2.Tables[0].Rows[index].ItemArray[0].ToString());

string select\_code\_command\_area = "SELECT Kod\_uch FROM dbo.Zem\_uch";

OleDbCommand command\_area\_select\_code = new OleDbCommand(select\_code\_command\_area, conn);

OleDbDataAdapter da3 = new OleDbDataAdapter(command\_area\_select\_code);

DataSet ds3 = new DataSet();

da3.Fill(ds3);

command\_declaration.Parameters.AddWithValue("Kod\_uch", ds3.Tables[0].Rows.Count);

command\_declaration.Parameters.AddWithValue("God", year);

command\_declaration.Parameters.AddWithValue("Kontr\_organ", tax\_insp);

command\_declaration.Parameters.AddWithValue("Podp1", sign1);

command\_declaration.Parameters.AddWithValue("Podp2", sign2);

command\_declaration.ExecuteNonQuery();

}

}

}

**ПРИЛОЖЕНИЕ Е  
Листинг программного модуля «SpecifiedDeclaration.cs»**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Data;

using System.Threading.Tasks;

using System.Data.OleDb;

using System.IO;

using Microsoft.Office.Interop.Excel;

using System.Windows.Forms;

namespace Diplom

{

class SpecifiedDeclaration

{

private float count\_prev\_decl;// { get; set; }

private float count\_after;// { get; set; }

private float change\_size;// { get; set; }

private float sur;// { get; set; }//штраф

private float fine;// { get; set; }//пеня

private string filePath;

private int area\_code;

private int obz\_code;

public SpecifiedDeclaration(float cur\_count\_prev\_decl, float cur\_count\_after, string cur\_filePath, int cur\_area\_code)

{

this.area\_code = cur\_area\_code;

this.count\_prev\_decl = cur\_count\_prev\_decl;

this.count\_after = cur\_count\_after;

this.change\_size = count\_after - count\_prev\_decl;

this.sur = (change\_size / 100) \* 3;

this.fine = (float)((change\_size / 100) \* 19 \* 1.2);

this.filePath = cur\_filePath;

export();

}

public void export()

{

Microsoft.Office.Interop.Excel.Application ObjExcel = new Microsoft.Office.Interop.Excel.Application();

Workbook ObjWorkBook = ObjExcel.Workbooks.Open(filePath, 0, false, 5, "", "", false, XlPlatform.xlWindows, "", true, false, 0, true, false, false);

Worksheet ObjWorkSheet = (Worksheet)ObjWorkBook.Sheets[1];

Range cell = ObjWorkSheet.get\_Range("D12");

cell.Value = count\_prev\_decl;

if(count\_after > count\_prev\_decl)

{

cell = ObjWorkSheet.get\_Range("D14");

cell.Value = count\_after;

cell = ObjWorkSheet.get\_Range("D18");

cell.Value = change\_size;

cell = ObjWorkSheet.get\_Range("D22");

cell.Value = sur;

cell = ObjWorkSheet.get\_Range("D24");

cell.Value = fine;

}

else

{

cell = ObjWorkSheet.get\_Range("D16");

cell.Value = count\_after;

cell = ObjWorkSheet.get\_Range("D20");

cell.Value = change\_size \* -1;

cell = ObjWorkSheet.get\_Range("D22");

cell.Value = 0;

cell = ObjWorkSheet.get\_Range("D24");

cell.Value = 0;

}

ObjExcel.Quit();

}

public void sp\_dataBaseInsert(OleDbConnection conn)

{

string insert\_command\_str = "INSERT INTO dbo.Nalog\_obz(Count\_pred, Count\_do\_uvel, Count\_do\_umen, Razm\_uvel, Razm\_umen, Shtraf\_sum, Penya\_sum) VALUES (?,?,?,?,?,?,?)";

string update\_command\_str = "UPDATE dbo.Dekl SET Kod\_obz = ? WHERE Kod\_uch = ?";

OleDbCommand insert\_command = new OleDbCommand(insert\_command\_str, conn);

OleDbCommand update\_command = new OleDbCommand(update\_command\_str, conn);

insert\_command.Parameters.AddWithValue("Count\_pred", count\_prev\_decl);

if (count\_after > count\_prev\_decl)

{

insert\_command.Parameters.AddWithValue("Count\_do\_uvel", count\_after);

insert\_command.Parameters.AddWithValue("Count\_do\_umen", 0);

insert\_command.Parameters.AddWithValue("Razm\_uvel", change\_size);

insert\_command.Parameters.AddWithValue("Razm\_umen", 0);

insert\_command.Parameters.AddWithValue("Shtraf\_sum", sur);

insert\_command.Parameters.AddWithValue("Penya\_sum", fine);

}

else

{

insert\_command.Parameters.AddWithValue("Count\_do\_uvel", 0);

insert\_command.Parameters.AddWithValue("Count\_do\_umen", count\_after);

insert\_command.Parameters.AddWithValue("Razm\_uvel", 0);

insert\_command.Parameters.AddWithValue("Razm\_umen", change\_size\*-1);

insert\_command.Parameters.AddWithValue("Shtraf\_sum", 0);

insert\_command.Parameters.AddWithValue("Penya\_sum", 0);

}

insert\_command.ExecuteNonQuery();

string select\_command\_str = "SELECT Kod\_obz FROM dbo.Nalog\_obz";

OleDbCommand select\_command = new OleDbCommand(select\_command\_str, conn);

select\_command.ExecuteNonQuery();

OleDbDataAdapter da = new OleDbDataAdapter(select\_command);

DataSet ds = new DataSet();

da.Fill(ds);

obz\_code = Convert.ToInt32(ds.Tables[0].Rows[ds.Tables[0].Rows.Count-1].ItemArray[0].ToString());

update\_command.Parameters.Add(new OleDbParameter("@obz\_code", obz\_code));

update\_command.Parameters.Add(new OleDbParameter("@current\_area", area\_code));

update\_command.ExecuteNonQuery();

}

}

}

**ПРИЛОЖЕНИЕ Ж**

**Листинг программного модуля «Quittance.cs»**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Word = Microsoft.Office.Interop.Word;

namespace Diplom

{

class Quittance

{

private string company\_name;// { get; set; }

private string egrpou;// { get; set; }

private string decl\_code;// { get; set; }

private string tax\_insp\_name;// { get; set; }

private string decl\_sign1;// { get; set; }

private string decl\_sign2;// { get; set; }

private string decl\_year;// { get; set; }

private string quittance\_number;// { get; set; }

public Quittance(string cur\_quittance\_number, string cur\_company\_name, string cur\_egrpou, string cur\_decl\_code,

string cur\_tax\_insp\_name, string cur\_decl\_sign1, string cur\_decl\_sign2, string cur\_decl\_year)

{

this.quittance\_number = cur\_quittance\_number;

this.company\_name = cur\_company\_name;

this.egrpou = cur\_egrpou;

this.decl\_code = cur\_decl\_code;

this.tax\_insp\_name = cur\_tax\_insp\_name;

this.decl\_sign1 = cur\_decl\_sign1;

this.decl\_sign2 = cur\_decl\_sign2;

this.decl\_year = cur\_decl\_year;

Word.Application word\_app = new Word.Application();

Word.Document quittance\_doc = word\_app.Documents.Add();

Word.Paragraph para = quittance\_doc.Paragraphs.Add();

para.Range.Text = "Налоговая квитанция" + "\nПредприятие: " + cur\_egrpou + "\nНомер декларации: " + cur\_decl\_code + "\nПериод: " + cur\_decl\_year +

"\nКонтролирующий орган: " + cur\_tax\_insp\_name + "\nПодписи документов:" + "\nПервая - директор: " + cur\_decl\_sign1 + "\nВторая - гл. бухгалтер: " + cur\_decl\_sign2 +

"\nГосударственная фискальная служба Украины";

para.Range.InsertParagraphAfter();

string fileName = "Квитанция №" + cur\_quittance\_number;

quittance\_doc.SaveAs2(fileName, Word.WdSaveFormat.wdFormatDocument);

quittance\_doc.Close();

word\_app.Quit();

}

}

}

**ПРИЛОЖЕНИЕ И**

**Листинг программного модуля «ReportsForm.cs»**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

using Microsoft.Reporting.WinForms;

namespace Diplom

{

public partial class ReportForm : Form

{

private CompanyReport companyReport;

public OleDbConnection conn { get; set; }

public ReportForm()

{

InitializeComponent();

}

private void ReportForm\_Load(object sender, EventArgs e)

{

this.Zem\_uchTableAdapter.Fill(this.Diplom\_demoDataSet.Zem\_uch);

companyReport = new CompanyReport();

string companyName = "ООО ДТЕК Высоковольтные сети";

companyReport.reportDataSet = companyReport.getDataForReport(companyName, conn);

reportViewer.ProcessingMode = ProcessingMode.Local;

LocalReport localReport = reportViewer.LocalReport;

localReport.ReportPath = @"C:\Users\Ulthwe\Documents\Visual Studio 2013\Projects\Diplom\Diplom\Report.rdlc";

ReportDataSource rds = new ReportDataSource();

rds.Name = "ReportDataSet";

rds.Value = companyReport.reportDataSet.Tables["Company\_areas"];

localReport.DataSources.Add(rds);

reportViewer.LocalReport.DataSources.Clear();

reportViewer.LocalReport.DataSources.Add(rds);

reportViewer.LocalReport.ReportPath = @"C:\Users\Ulthwe\Documents\Visual Studio 2013\Projects\Diplom\Diplom\Report.rdlc";

reportViewer.RefreshReport();

this.reportViewer.RefreshReport();

}

private void reportViewer\_Load(object sender, EventArgs e)

{

}

}

}

**ПРИЛОЖЕНИЕ К**

**Листинг программного модуля «CompanyReport.cs»**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.OleDb;

namespace Diplom

{

class CompanyReport

{

public DataSet reportDataSet { get; set; }

public CompanyReport()

{

}

public DataSet getDataForReport(string companyName, OleDbConnection conn)

{

string report\_query\_command\_str = "SELECT Name AS [Наименование], " +

"Ploshad AS [Площадь], " +

"Sum\_lgoti AS [Сумма льготы], " +

"God\_sum\_pay AS [Годовая сумма налога (к оплате)] " +

"FROM dbo.Yur\_litso YL LEFT JOIN dbo.Dekl DK ON YL.Kod\_litsa = DK.Kod\_litsa LEFT JOIN dbo.Zem\_uch ZU ON DK.Kod\_uch = ZU.Kod\_uch " +

"WHERE Name = ?";

//string report\_query\_command\_str = "SELECT \* FROM dbo.Zem\_uch";

OleDbCommand report\_command = new OleDbCommand(report\_query\_command\_str, conn);

report\_command.Parameters.Add(new OleDbParameter("@companyName", companyName));

report\_command.ExecuteNonQuery();

reportDataSet = new DataSet();

OleDbDataAdapter report\_data\_adapter = new OleDbDataAdapter(report\_command);

report\_data\_adapter.Fill(reportDataSet, "Company\_areas");

return reportDataSet;

}

}

}