Федеральное государственное бюджетное образовательное учреждение
высшего образования

«ПЕРМСКИЙ ГОСУДАРСТВЕННЫЙ ГУМАНИТАРНО-ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ»

ФАКУЛЬТЕТ ИНФОРМАТИКИ И ЭКОНОМИКИ

Кафедра прикладной информатики

Выпускная квалификационная работа

РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ «ДОКУМЕНТООБОРОТ КАЗНАЧЕЙСКОГО УЧРЕЖДЕНИЯ»

на основе архитектуры «клиент-сервер»

Работу выполнил:

студент Z1253Э группы

направления подготовки 09.03.03. «Прикладная информатика», профиль «Прикладная информатика в экономике»

Симаков Николай Александрович

_____

(подпись)

«Допущен к защите в ГАК»

Зав. кафедрой

_____

(подпись)

«____» _____ 2016 г.

Руководитель:
кандидат тех. наук, доцент кафедры прикладной информатики
Сичинава Зураби Иродиевич

_____

(подпись)

ПЕРМЬ
2016

Оглавление

Введение

В обществе вопрос обработки документов и их хранения с каждым годом ,за последние несколько десятилетий, увеличивал свою значимость и влияние на жизнь в окружающей среде. Это явление затрагивает не только жизнедеятельность человека, но и те аспекты, которые считаются жизненно важными для него. Условия развития современного и ритмичного бизнеса, тем самым, заставляют многие компании прибегать к внедрению специализированных систем, которые позволяют увеличить скорость и уровень эффективности бизнес-процессов в организациях.

Сейчас, в быстро меняющихся условиях деловой среды, время является важнейшим показателем эффективности в управлении процессами на предприятии. Так и в случае с информационными потоками, регулирующими документальные аспекты деятельности организации, существует необходимость в автоматизации их обработки. Таким образом, использование компьютерных информационных систем для обеспечения высокого уровня производительности, является прерогативным направлением развития обработки поступающей информации в наше время.

Объектом исследования является деятельность служб «Управления финансов и налоговой политики администрации Гремячинского муниципального района», осуществляющих регистрацию и поиск документов.

Предметом исследования является автоматизация процессов регистрации и поиска документов, поступивших и созданных в организации.

Целью данной выпускной работы является разработка информационной системы документооборота организации на основе технологии «клиент-сервер». Для достижения поставленной цели необходимо решить следующие задачи:

1. Проанализировать предметную область документооборота;

2. Изучить процесс движения документов в организации;
3. Произвести концептуальное проектирование;
4. Спроектировать пользовательский интерфейс информационной системы
5. Реализовать информационную систему;
6. Рассчитать экономическую эффективность от внедрения информационной системы документооборота.

Работа состоит из введения, 3 глав, заключения, списка используемой литературы и приложения.

В первой главе будет проведен анализ предметной области, сравнительный обзор программных продуктов.

Разработка проектируемого программного продукта будет рассмотрена во второй главе.

В заключительной, третьей главе, произведем анализ экономической эффективности.

1. Аналитический обзор

1.1. Анализ предметной области.

Рассмотрим главный объект делопроизводства – документ. Документ – это материальный объект, содержащий информацию в зафиксированном виде и специально предназначенный для её передачи во времени и пространстве.[1]

Так же Федеральный закон № 77-ФЗ «Об обязательном экземпляре документов» говорит, что документ – это материальный носитель с зафиксированной на нём в любой форме информацией в виде текста, звукозаписи, изображения и (или) их сочетания, который имеет реквизиты, позволяющие его идентифицировать, и предназначен для передачи во времени и в пространстве в целях общественного использования и хранения.

В современном мире работа с различными видами документов в организациях определена инструкциями, определенных в соответствующем положении. Движение документов на предприятиях так же определяется положением о документообороте. Для контроля работы сотрудников с корреспонденцией существует подход, суть которого заключается введении картотеки с фиксированием в карточках всех этапов жизни документа на предприятии: от поступления и до сдачи его в архив.[2] Ранее такие картотеки существовали в бумажном варианте, а сейчас – в электронном.

Сущность электронного документоведения заключается в совокупности автоматизированных процессов по работе с документами, представленными в электронном виде, с реализацией концепции «безбумажного делопроизводства».[3] Оно основано на следующих принципах:

1. Понятие электронного документооборота включает как внутренний, так и внешний документооборот

2. Стирание грани между техническими и содержательными операциями при подготовке документов и в процессе документооборота при использовании современных информационных технологий
3. Сотрудники организации (включая специалистов и руководителей) становятся непосредственными участниками электронного документооборота в рамках деловых и управленческих процессов
4. Оптимизация деловых и управленческих процессов, маршрутизация документов в организации на основе корпоративных информационных технологий, возможность одновременной работы над электронным документом несколькими участниками документооборота
5. Регистрация в СЭД всех основных категорий документов организации, в том числе внутренних, а также учет проектов документов и применение процедур электронного согласования
6. Однократная регистрация документов в СЭД
7. Ведение единой регистрационной базы поступающих и отправляемых (входящих, исходящих и внутренних) документов с возможностью децентрализации приема документов, их регистрации и отправки
8. Документационная служба организует управление документацией и документооборотом в организации, выполняя при этом те задачи и виды работ с документами, которые требуют централизации.[4]

В «Управлении финансов и налоговой политики администрации Гремячинского муниципального района» документооборот построен следующим образом: вся входящая, исходящая и внутренняя корреспонденция

регистрируется делопроизводителем на бумажном носителе, представленном в виде журналов.

Всего журналов для учета поступившей, отправленной и внутренней корреспонденции три:

1. Переписка с разными министерствами;
2. Переписка с Министерством финансов;
3. Переписка с разными организациями и учреждениями.

Доведение входящей корреспонденции до исполнителя производится путем печати документа и его вручения на обработку. Отправление документов после исполнения выполняется путем его регистрации в журнале и отправкой посредством электронной или обычной почты, в зависимости от вида корреспонденции. Внутренние документы, как правило, не регистрируются, а подшиваются в личное дело сотрудника, в отношении которого были созданы.

Такой метод обработки документов существует с момента образования отдела и у него имеется один существенный недостаток – это затруднительный поиск какого-либо документа по определенным параметрам, будь то исполнитель или отправитель, либо тема документа. Этот недостаток выражается во временных потерях занятости сотрудников. На данном временном этапе развития общества такой подход к ведению делопроизводства считается анахронизмом.

Существует необходимость в автоматизации данного процесса при помощи современных подходов к информационным потокам в УФиНП, а именно создании информационной среды для решения основных задач оборота документов, тем самым сократив временные расходы на работу с ними.

В сложившейся ситуации руководителем УФиНП было принято решение по автоматизации данного процесса, путем внедрения какой-либо

информационной системы (ИС) для работы с документами из существующих на рынке программного обеспечения, либо создания своей ИС.

1.2. Сравнительный анализ ИС для документооборота

В процессе изучения сегмента рынка программного обеспечения, решающие задачу оборота документов на предприятии были отобраны несколько ИС для решения существующей проблемы в организации.

DIRECTUM – система электронного документооборота и управления взаимодействием, нацеленная на повышение эффективности работы всех сотрудников организации в разных областях совместной деятельности.

Система Docsvision 5 предназначена для автоматизации управления документами и бизнес-процессами, включая как общую управленческую деятельность, так и различные функциональные задачи подразделений, и операционные процессы в деятельности предприятий.

«1С:Документооборот» – система электронного документооборота (СЭД), созданная фирмой 1С на основе платформы 1С:Предприятие и обеспечивающая полный цикл обработки документов компании, в комплексе решающий задачи автоматизации учета документов, взаимодействия сотрудников, контроля и анализа исполнительской дисциплины.

Сравнительные характеристики указанных ИС и их стоимость, из расчета использования 10 сотрудниками, приведены в таблице 1.

Таблица №1

Критерии	Docsvision	Directum	1С: Документооборот
Общее			
ОС сервера	MS Windows Server 2003, 2008, 2008 R2	Microsoft Windows Server 2003, 2008, 2008 R2, 2012	Windows Server 2003, 2008, 2012. Windows XP, Vista, 7, 8. Linux
ОС клиента	Минимальные требования: MicrosoftWindows XP Professional (SP3 и выше); MicrosoftWindows 7 (любая редакция). Рекомендуется: MicrosoftWindows 7 (любая редакция).	Microsoft Windows XP Professional Microsoft Microsoft Windows 7 Professional/Enterprise/Ultimate 32-разрядная или 64-разрядная Microsoft Windows 8 Pro/Enterprise 32-разрядная или 64-разрядная	Windows Server 2003, 2008, 2008 R2, 2012. Windows XP, Vista, 7, 8. Linux, MacOSX 10.5 и выше. Мобильный клиент: iOS 3.2, 4.2 и выше (только планшеты iPad), Android.
Тип клиентского приложения	•Толстый клиент; •Тонкий клиент; •PM Руководителя (толстый клиент). •Мобильные клиенты под iPad и Android. • Клиент под Windows 8	•Толстый клиент; •Тонкий клиент (в том числе для мобильных устройств); •DIRECTUM Solo (Клиент для планшетов и ноутбуков на ОС Windows 7, 8); •DIRECTUM для SharePoint; •Клиентпод iPad.	•Толстый клиент; •Тонкий клиент; •Web-клиент; •Мобильный клиент для iOS и Android (тестовая версия).
Web-браузер	Microsoft Internet Explorer 8.0 и выше Google Chrome Safari	Internet Explorer, Firefox, Opera, включая браузеры мобильных устройств (Opera Mobile, Safari Mobile, IE Mobile)	Microsoft Internet Explorer 8.0, 9.0(x86), 10(x86). Mozilla Firefox от 17, Google Chrome от 4, Safari 4.0.5 и выше.

СУБД	MS SQL Server (2005, 2008, 2008 R2, 2012).	MS SQL Server (2005, 2008, 2008 R2).	Microsoft SQL Server (2000, -2012). PostgreSQL (8.1.5 - 9.2.4). IBM DB2 (9.1 - 10.1). Oracle Database (10gR2, 11gRl, llgR2).
Поиск документов	По реквизитам, полнотекстовый, с учетом морфологии.	По реквизитам, по штрих-коду, полнотекстовый.	По реквизитам, полнотекстовый, с учетом русской, английской и украинской морфологии, по штрих-коду.
Обработка документов различных типов			
Типы документов	Входящий, исходящий, внутренний, документ. Есть возможность добавлять собственные виды документов	В стандартную поставку системы входит около 30 видов документов, под которые есть карточки.	Обращения граждан, входящий, исходящий, внутренний, договорной документы.
Обеспечение информационной безопасности			
Разграничение прав доступа	• Каждому пользователю и каждому объекту назначается уровень допуска; •Используется ролевая модель.	Права назначаются на уровне пользователей и групп пользователей как на компоненты системы в целом, так и на отдельные записи и реквизиты.	Используются ограничения на уровне ролей и ограничения на уровне записей базы данных (RLS - RecordLevelSercurity) платформы «1С».
Стоимость СЭД			
Правила лицензирования	Лицензии конкурентные (количество одновременных подключений).	Лицензии могут быть именными или конкурентными (в зависимости от редакции).	Лицензии конкурентные (количество одновременных подключений).
Стоимость на подключение	23000	59000	77400

10 пользователей (руб.)			
Стоимость технической поддержки на 1 год от производителя СЭД. (руб.)	0	0	19776
Стоимость технической поддержки на 2 и последующие годы от производителя СЭД. (руб.)	4600	11800	29664

Исходя из полученных данных, следует, что информационные системы обладают следующими признаками:

1. Достаточный функционал всех перечисленных ИС;

2. Docsvision и DIRECTUM используют проприетарные СУБД;

3. DIRECTUM и «1С:Документооборот 8» обладают высокой стоимостью владения.

Выводы

Как мы можем увидеть из приведенных выше кратких характеристик ИС документооборота, все они обладают достаточным спектром предоставляемых действий с документами. Так же все они имеют закрытый код, что затрудняет адаптацию к бизнес-процессам компании и интеграцию с другими подсистемами. И обладают существенной функциональной избыточностью, которая не будет применяться в условиях организации.

На основании аналитического обзора, необходимых для исследования факторов, руководителем «Управления финансов и налоговой политики администрации Гремячинского района» было принято решение о разработке собственной ИС для обработки документов.

2. Разработка информационной системы

2.1. Проектирование ИС

Суть проектирования информационной системы заключается в описании деятельности организации, концептуальном проектировании, проектировании пользовательского интерфейса и разработке программной составляющей ИС.

2.1.1. Описание структуры «Управления финансов и налоговой политики администрации Гремячинского муниципального района»

Общая характеристика

Управление финансов и налоговой политики является структурным подразделением администрации Гремячинского муниципального района и входит в систему органов местного самоуправления. УФиНП осуществляет управление в сфере бюджетного процесса, исполнение бюджета Гремячинского муниципального района и консолидированного бюджета района, контроль соблюдения бюджетного законодательства главными распорядителями, распорядителями, получателями средств местного бюджета. Отдел является юридическим лицом, имеет печать с реквизитами, штамп, бланки со своим наименованием. Имеет самостоятельный баланс, счет в банке Российской Федерации, лицевые счета в органах казначейства.

Организационная структура

Штат Управления финансов и налоговой политики состоит из 13 человек. Управление возглавляет начальник, назначаемый и освобождаемый от должности главой муниципального образования. Начальник Управления финансов осуществляет руководство на принципе единоначалия и несет персональную ответственность за выполнение возложенных на финансовое управление задач и осуществление им своих функций.

Прием и увольнение муниципальных служащих управления, их поощрение (за исключением премирования по результатам работы) и применение к ним мер дисциплинарного воздействия осуществляет начальник Управления.

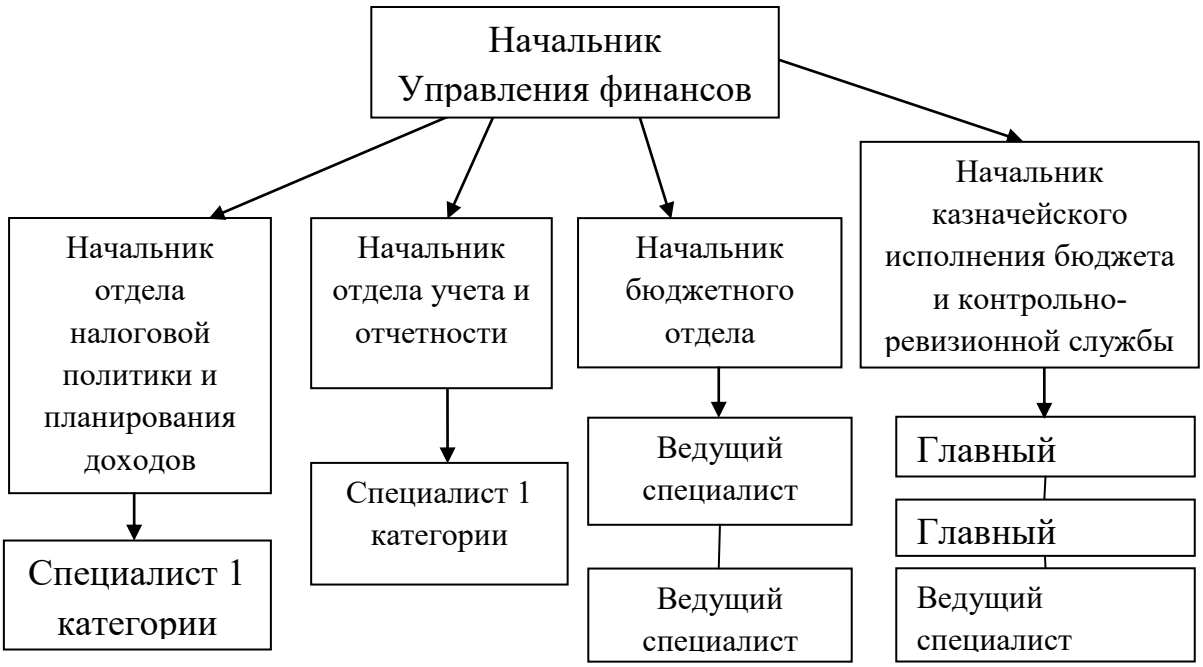

Рис. 1 Структура управления финансов и налоговой политики администрации Гремячинского муниципального района

Управление деятельностью

Основными задачами Управления финансов и налоговой политики является решение вопросов местного значения муниципального района:

- разработка и осуществление финансовых планов и мероприятий в отраслях хозяйства Гремячинского муниципального района в целях повышения его эффективности и рационального использования бюджетных средств.
- составление бюджета Гремячинского муниципального района в соответствии с прогнозами поступлений доходов и программами, одобренными органами представительной и исполнительной власти,

обеспечивающими единую социально-экономическую и налоговую политику, проводимую в Гремячинском районе.

- подготовка предложений по совершенствованию межбюджетных отношений, совершенствованию налоговой политики, а также предложений о введении местных налогов и сборов.
- осуществление финансового контроля над сохранностью, эффективным и целевым использованием средств, выделенных из бюджета Пермского края.
- ведение учета и составление отчетности об исполнении бюджета Гремячинского муниципального района и консолидированного бюджета района, а также осуществление контроля над состоянием учета и отчетности в органах местного самоуправления, бюджетных учреждениях, финансируемых из местного бюджета.

Управление в соответствии с возложенными на него задачами осуществляет следующие основные функции:

1. В области составления бюджета Гремячинского муниципального района;
2. В области инвестиционной политики;
3. В области совершенствования межбюджетных отношений, совершенствования налоговой политики, а также введения местных налогов и сборов;
4. В области контрольно-экономической работы;

Отделы УФиНП администрации Гремячинского муниципального района

Управление финансов состоит из четырех отделов: бюджетный отдел, отдел налоговой политики и планирования доходов, отдел учета и отчетности, отдел казначейского исполнения бюджета и контрольно-ревизионной службы.

1. Бюджетный отдел.

Основным документом для работы бюджетного отдела является Бюджетный кодекс РФ. Он служит целям финансового регулирования,

устанавливает общие принципы бюджетного законодательства РФ, правовые основы функционирования бюджетной системы РФ, правовое положение субъектов бюджетных правоотношений, порядок регулирования межбюджетных отношений, определяет основы бюджетного процесса в РФ, основания и виды ответственности за нарушение бюджетного законодательства Российской Федерации.

2. Отдел налоговой политики и планирования доходов.

Основным документом для работы отдела налоговой политики и планирования доходов является налоговый кодекс РФ. Отдел в течение года ведет роспись доходов, вносит изменения в роспись на основании решения Земского Собрания (перевыполнение собственных доходов, поступления из краевого бюджета).

3. Отдел учета и отчетности.

Бухгалтерский учет в Управлении финансов ведется на основании Закона о бухгалтерском учете №402-ФЗ, учетной политики, утвержденной приказом № 2 от 10 января 2013 года. Ведение бухгалтерского учета осуществляется отделом учета и отчетности. Ответственность за организацию бухгалтерского учета возлагается на начальника отдела - главного бухгалтера.

4. Отдел казначейского исполнения бюджета и контрольно-ревизионной службы.

Основной целью казначейского исполнения бюджета является оптимизация процесса исполнения бюджета, повышение эффективности использования средств местного бюджета.

Специалист по контрольно-ревизионной работе осуществляет контроль за целевым использованием бюджетных средств различного уровня, организует подготовку и проведение контрольных мероприятий, разрабатывает программы и планы ревизий, проверок, используя действующий законодательный и инструктивный материал. Готовит и направляет по результатам ревизий и проверок начальнику Управления финансов справки, акты и заключения со

своими предложениями, направленными на устранение нарушений и недостатков.

2.1.2. Концептуальное проектирование

Сущность концептуального проектирования заключается в проверке общей структуры и учета в ней всех аспектов решаемой задачи, а именно:

- выделение объектов предметной области и их атрибутов;
- определение связей между объектами и построение ER-диаграммы.

Функциональное проектирование

На основании описания деятельности УФиНП следует выделить основные бизнес-процессы документооборота:

1. Входящий поток:
    а) Получение корреспонденции;
    б) Обработка;
    в) Регистрация;
    г) Уведомление руководителя.
2. Исходящий поток:
    а) Получение задания;
    б) Контроль исполнения;
    в) Регистрация документа;
    г) Отправка адресату.
3. Внутренний поток:
    а) Поручение задачи;
    б) Обсуждение исполнения;
    в) Выполнение;
    г) Прикрепление к делу исполнителя.

Проектирование существующих бизнес-процессов осуществляется по методологии SADT и её нотациях IDEF0, IDEF3, DFD реализуемые, как AS-IS и TO-BE в программных средствах Erwin Process Modeler 7.3 и Rational Rose 2003.

На диаграмме «Дерево узлов» (рис. 2)отображена иерархия, позволяющая увидеть всю модель в целом. Данная диаграмма построена в нотации DFD.

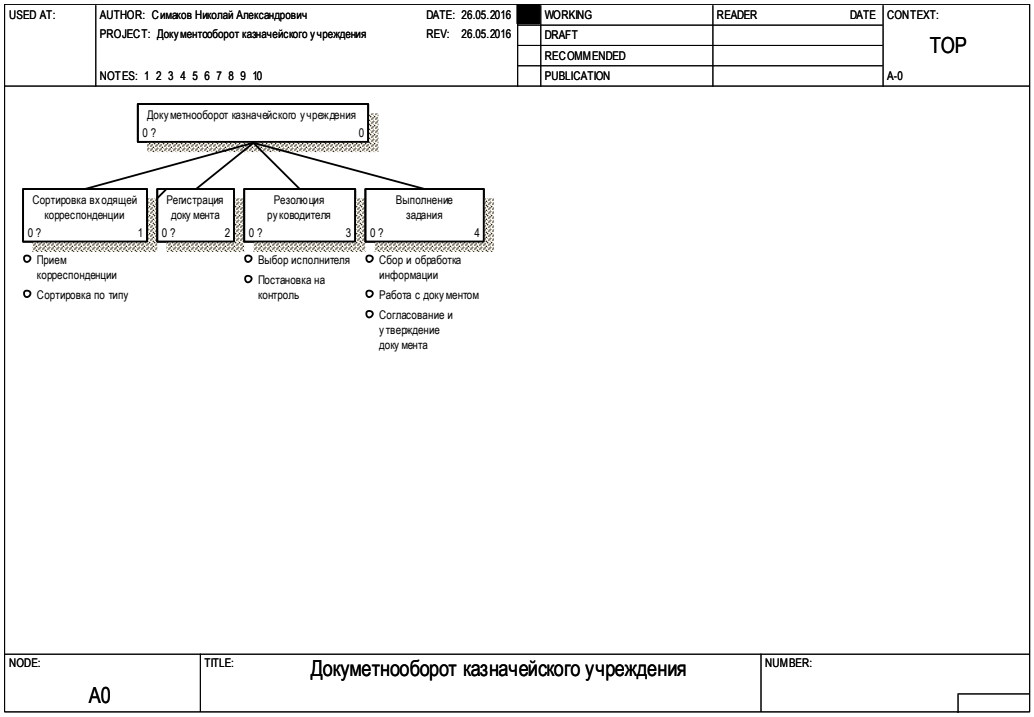

Рисунок 2. Дерево узлов документооборота УФиНП

На рисунке № 3 отражена контекстная диаграмма документооборота в организации, выполненной в нотации IDEF0.

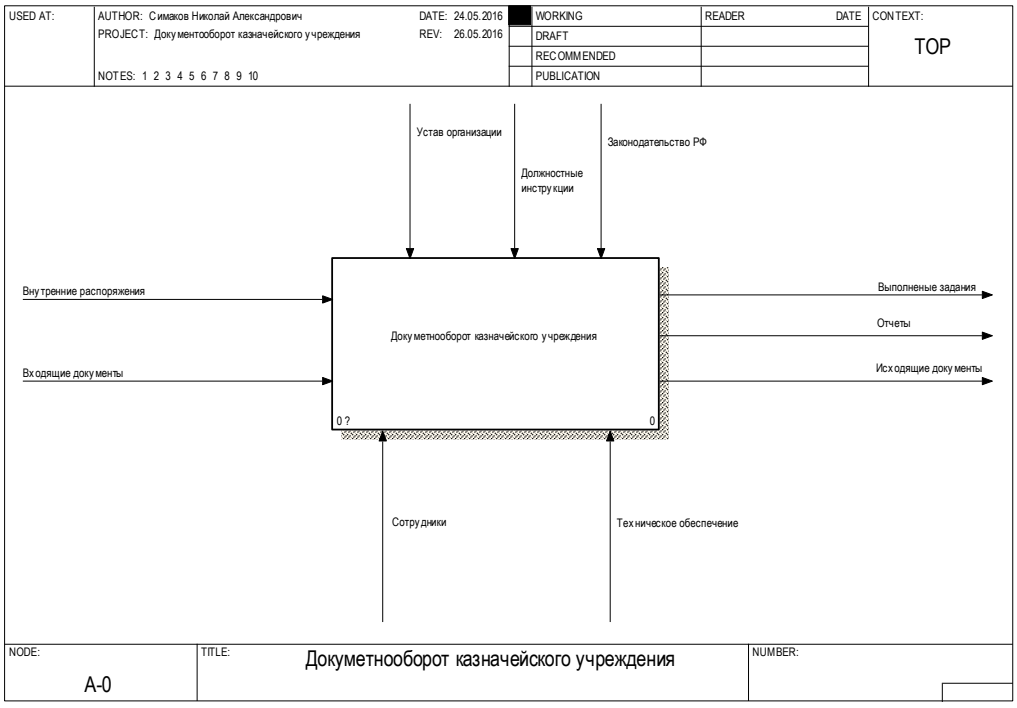

Рисунок 3. Контекстная диаграмма документооборота УФиНП

Далее раскрываем суть документооборота в УФиНП декомпозицией контекстной диаграммы (рис. 4), отображая на нем основные бизнес-процессы.

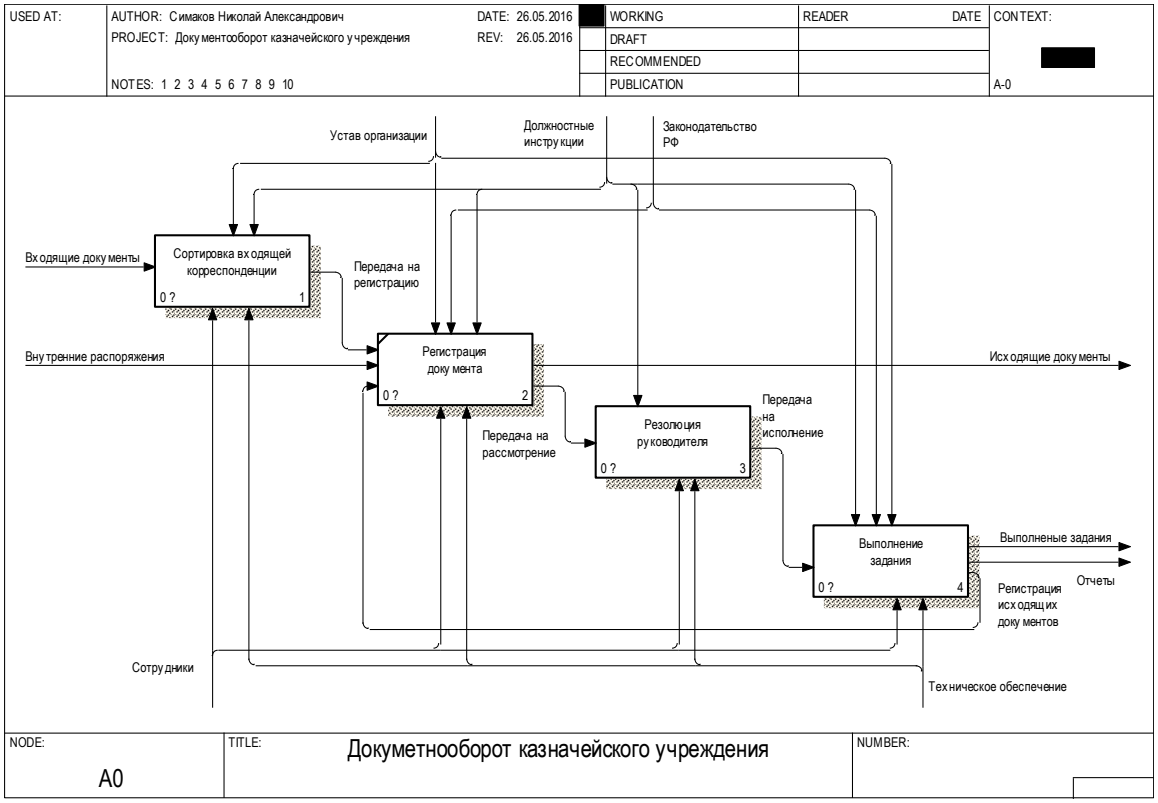

Рисунок 4. Декомпозиция контекстной диаграммы оборота документов УФиНП

Продолжим раскрытие декомпозиции контекстной диаграммы в соответствии с диаграммой «Дерево узлов» (рис.1) для декомпозиции соответствующего бизнес-процесса (рис. 5 – 7).

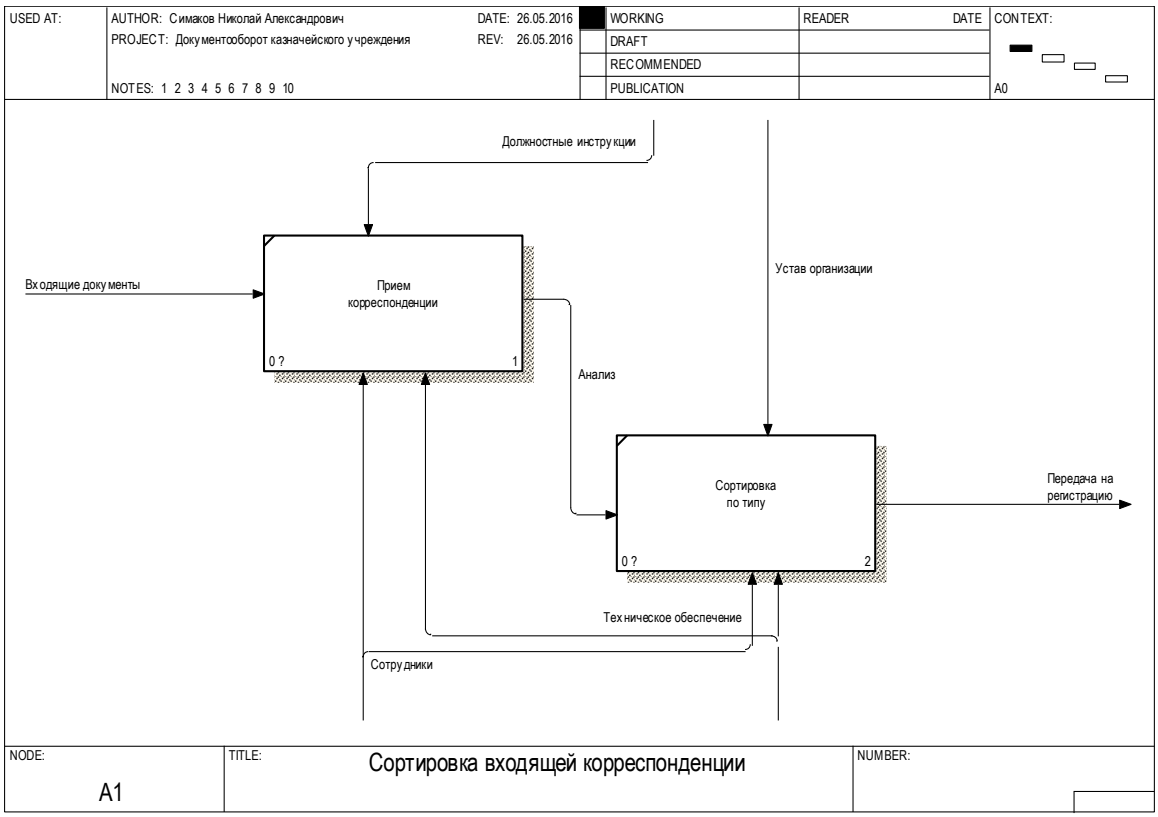

Рисунок 5. Декомпозиция бизнес-процесса «Сортировка входящей корреспонденции»

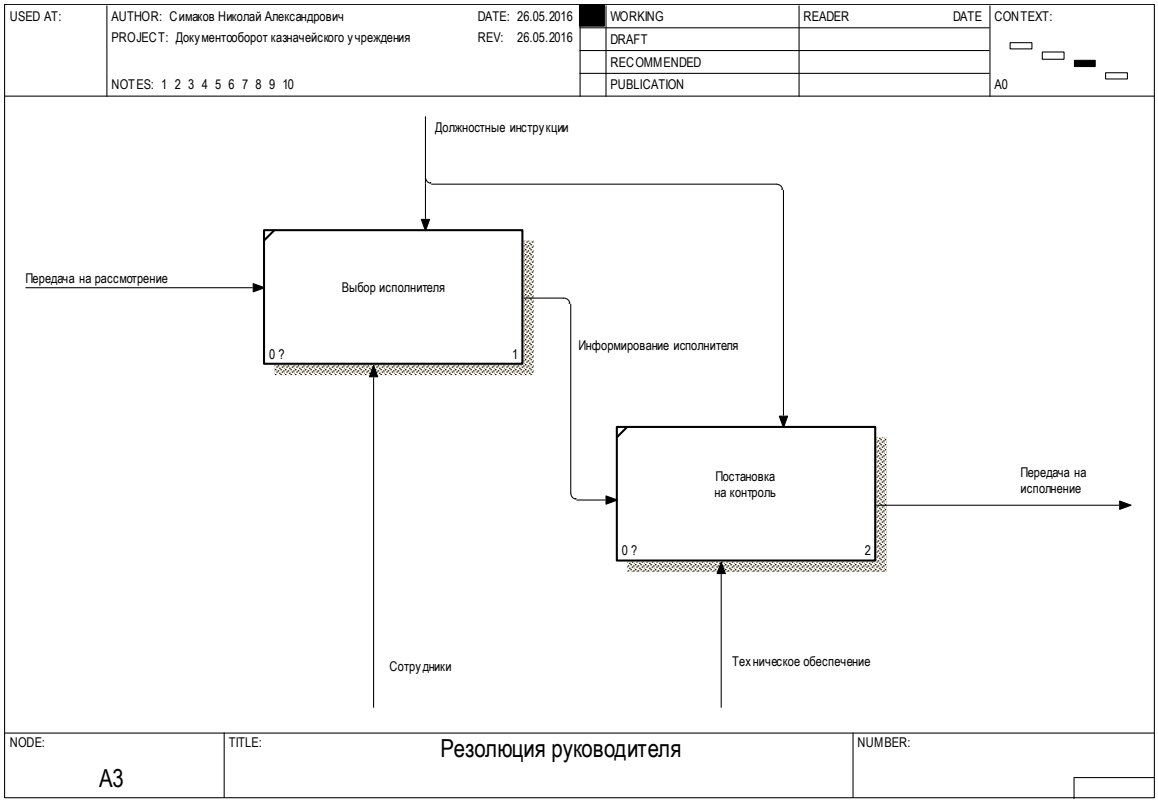

Рисунок 6. Декомпозиция бизнес-процесса «Резолюция руководителя»

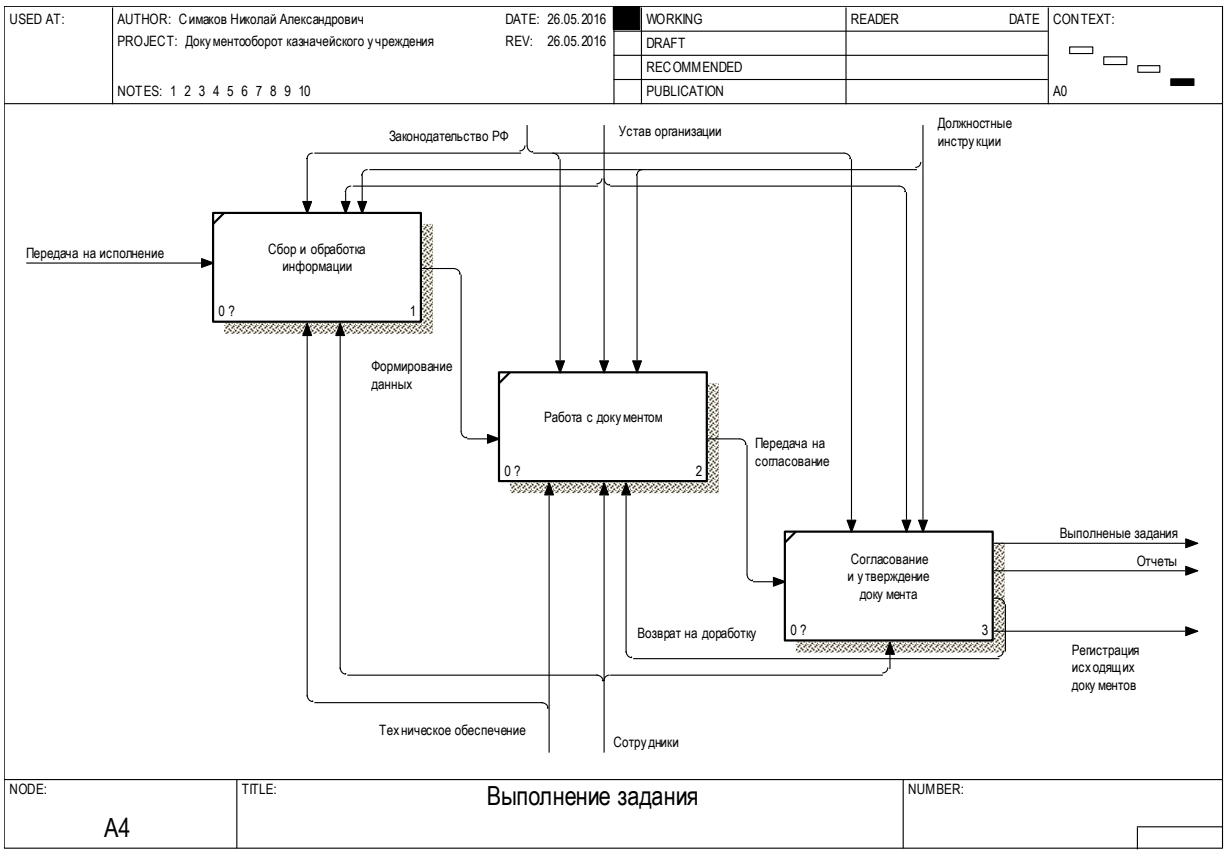

Рисунок 7. Декомпозиция бизнес-процесса «Выполнение задания»

В дальнейшем, на основании описания предметной области и анализа спроектированной модели AS-IS существующей организации процесса документооборота УФиНП необходимо создать модель TO-BE, оптимизируя работу с документами. Произведем аналогичные действия, произведя подобные действия, как в случае с проектированием модели AS-AS.

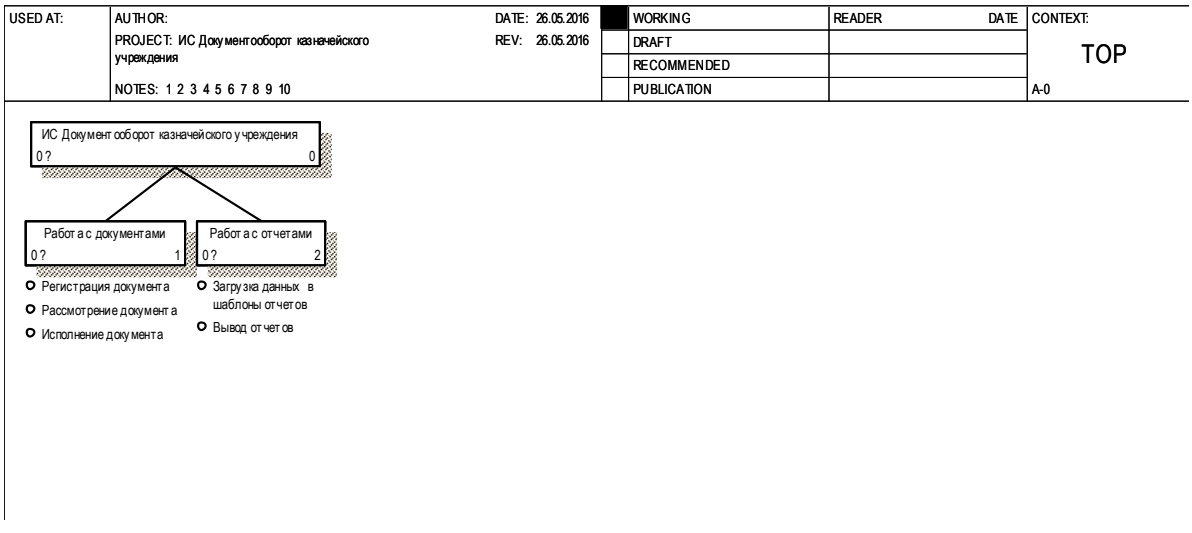

Рисунок 8. Дерево узлов ИС Документооборот казначейского учреждения

Как видим из вновь созданного дерева узлов (рис. 8) информационной системы документооборота УФиНП основные бизнес-процессы преобразованы и сокращены до двух, тем самым сократив объем действий с документами.

В контекстной диаграмме так же произошло сокращение выполненных заданий, соединив их в единый поток с исходящими документами (рис. 9).

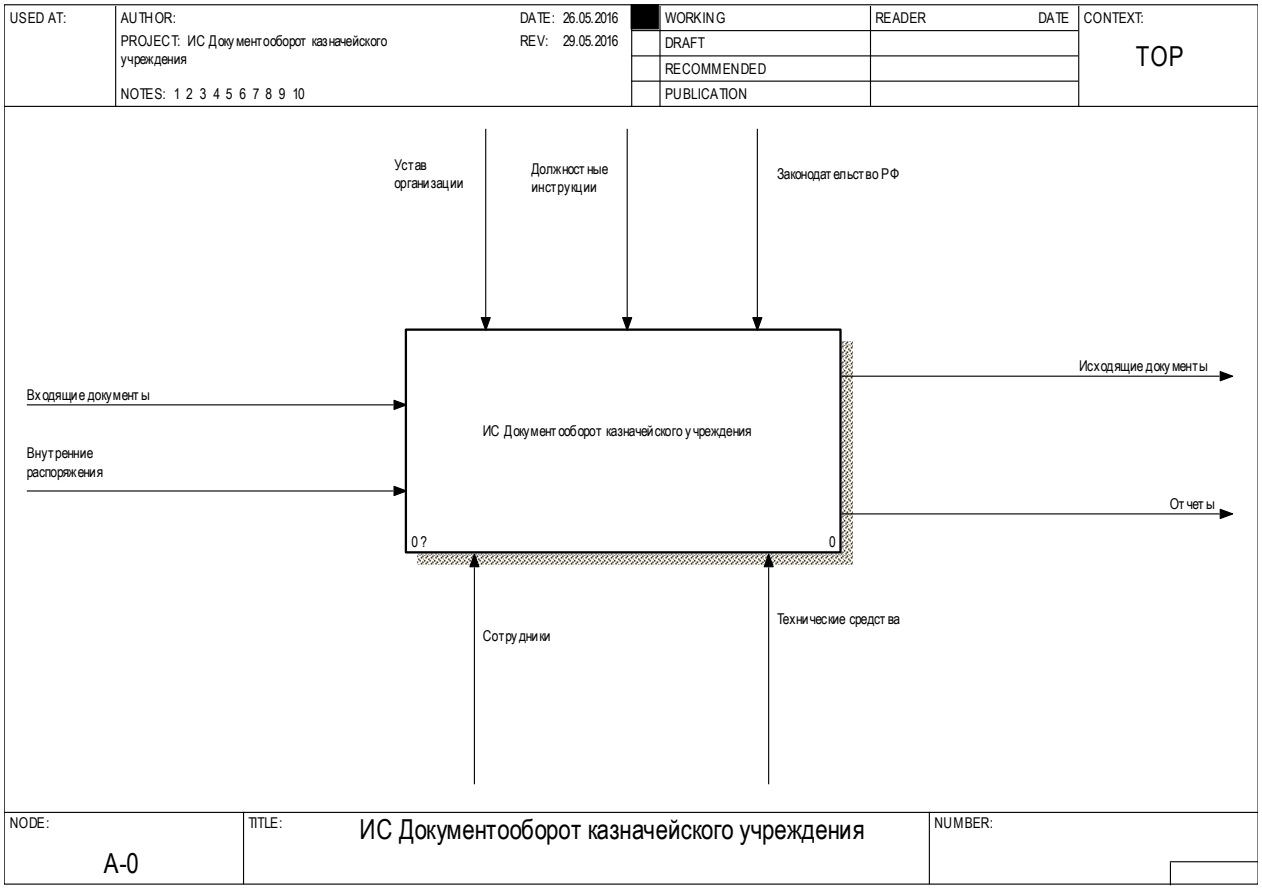

Рисунок 9. Контекстная диаграмма ИС Документооборот казначейского учреждения

Далее продолжаем раскрывать работу с документами в проектируемой ИС «Документооборот казначейского учреждения». В декомпозиции контекстной диаграммы ИС (рис. 10) отображены только два основных бизнес-процесса по работе с документами, как и было сказано о диаграмме «Дерево узлов»(рис. 8). На рисунках 11 и 12 изображены декомпозиции «Работа с документами» и «Работа с отчетами» соответсвенно.

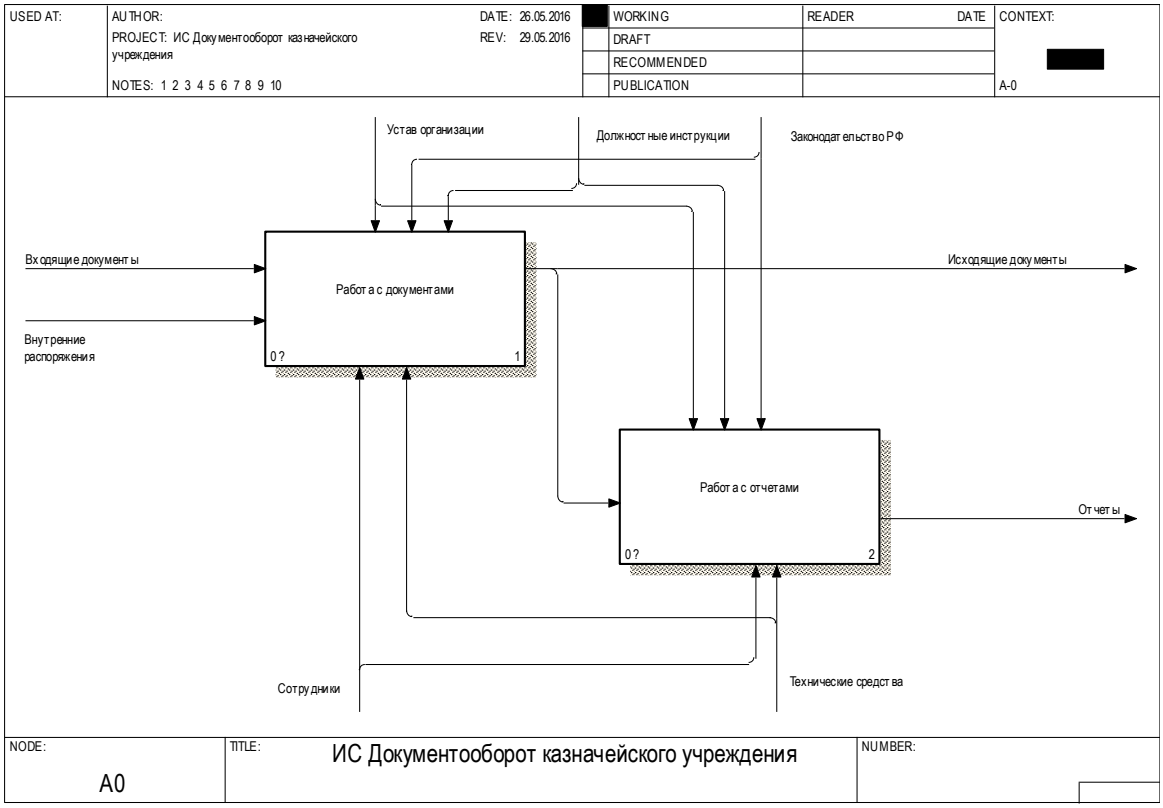

Рисунок 10. Декомпозиция контекстной диаграммы ИС Документооборот казначейского учреждения

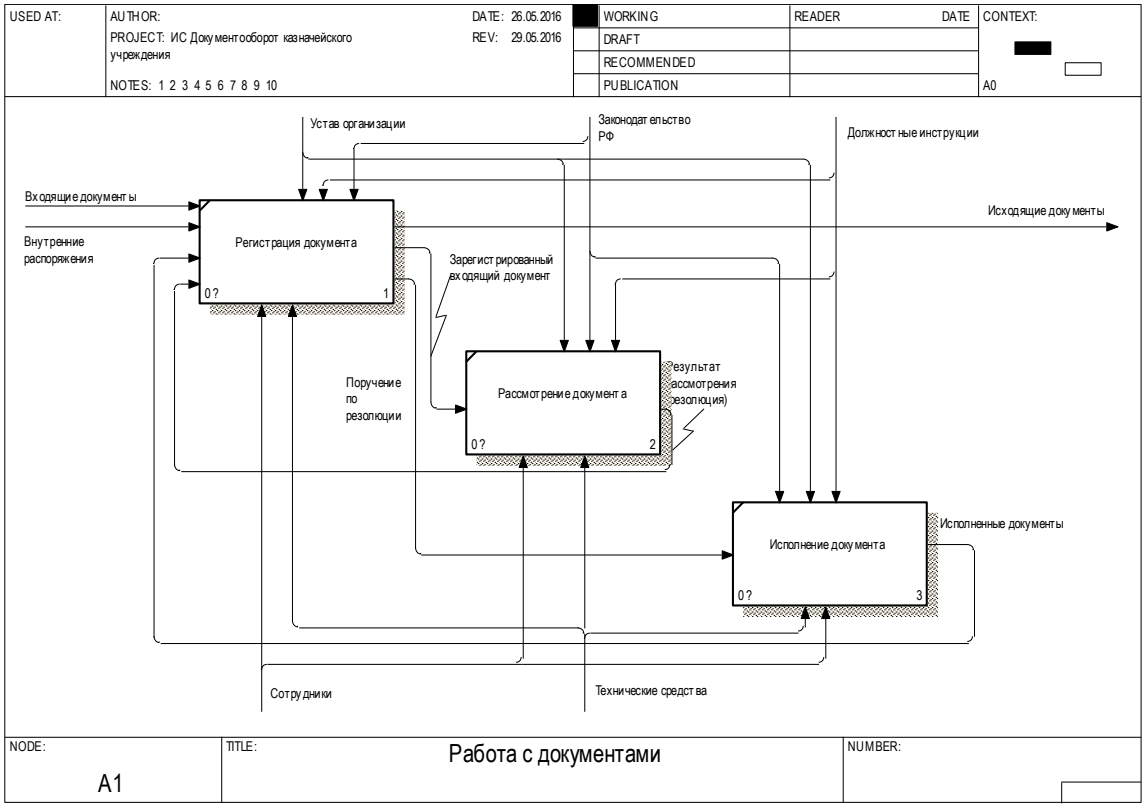

Рисунок 11. Декомпозиция бизнес-процесса «Работа с документами»

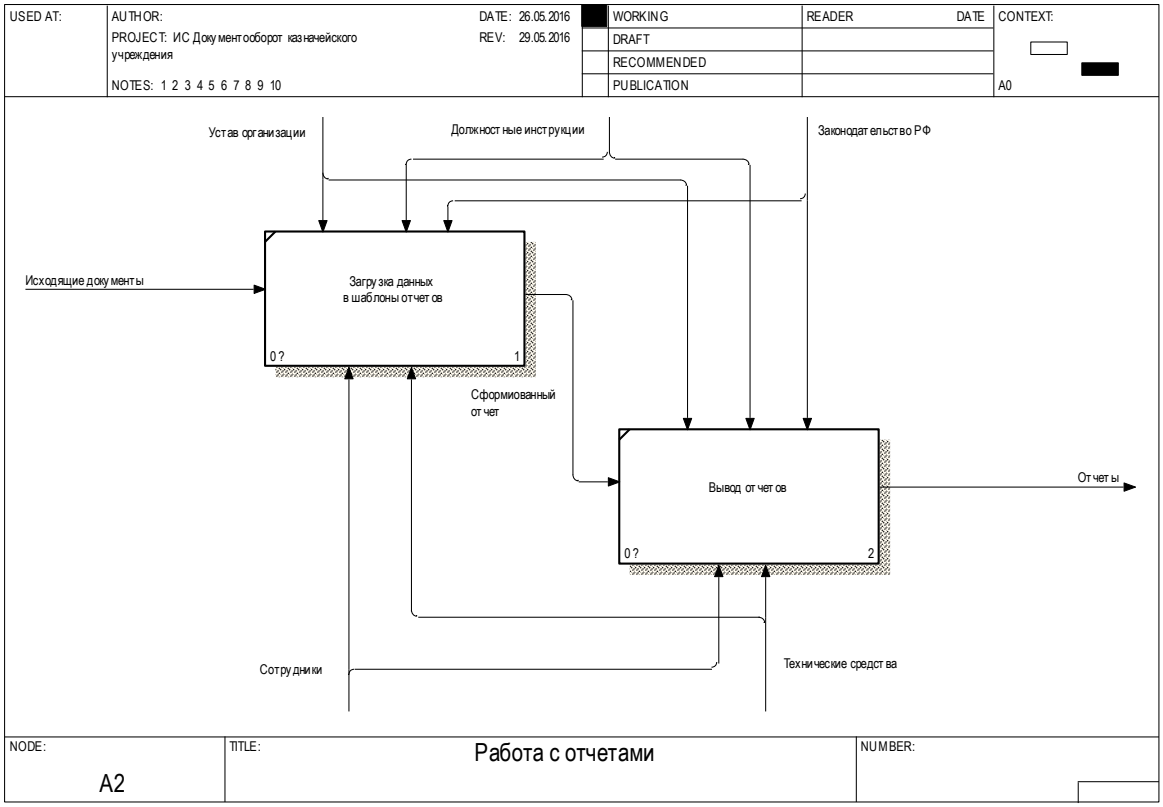

Рисунок 12. Декомпозиция бизнес-процесса «Работа с отчетами»

Объекты и атрибуты

На основании описания предметной области и функционального проектирования выделим следующие объекты и атрибуты для проектируемой информационной системы (табл. 2).

Таблица 2. Таблицы и атрибуты

Объект	**Атрибут**	**Первичный ключ**
Сотрудник	Код сотрудника Код подразделения Код должности Ф.И.О. Адрес Телефон	Код сотрудника
Подразделения	Код подразделения	Код подразделения

	Наименование	
Документ	Код документа Вид документа Тип документа Код организации Код сотрудника Номер документа Наименование Дата регистрации Дата отправки Вложение Примечание	Код документа
Организация	Код организации Наименование Адрес Телефон Электронная почта	Код организации
Должность	Код должности Наименование	Код должности

Связи

Далее определим связи между объектами, используя семантическую методологию (табл. 3).

Таблица 3. Связи

Название	Объекты	Кардинальность	Степень участия
ЗАНИМАЕТ	СОТРУДНИК ДОЛЖНОСТЬ	M:1	Полная Частичная
РАБОТАЕТ	СОТРУДНИК ОТДЕЛ	M:1	Полная Частичная
РЕГИСТРИРУЕТ	СОТРУДНИК ДОКУМЕНТ	M:1	Полная Полная
ИМЕЕТ	ДОКУМЕНТ ТИП ДОКУМЕНТА	M:1	Полная Частичная
ИМЕЕТ	ДОКУМЕНТ ВИД ДОКУМЕНТА	M:1	Полная Частичная
КОРРЕСПОНДИРУЕТ	ОРГАНИЗАЦИЯ ДОКУМЕНТ	M:1	Полная Частичная

ER-диаграмма

ER-диаграмма – этоEntity-Relationship диаграмма «сущность-связь» и представляет собой простой способ моделирования данных и отношений между ними с использованием стандартных символов. Создадим данную диаграмму, основываясь на определенных ранее объектах, их атрибутах и связях между ними (рис. 13).

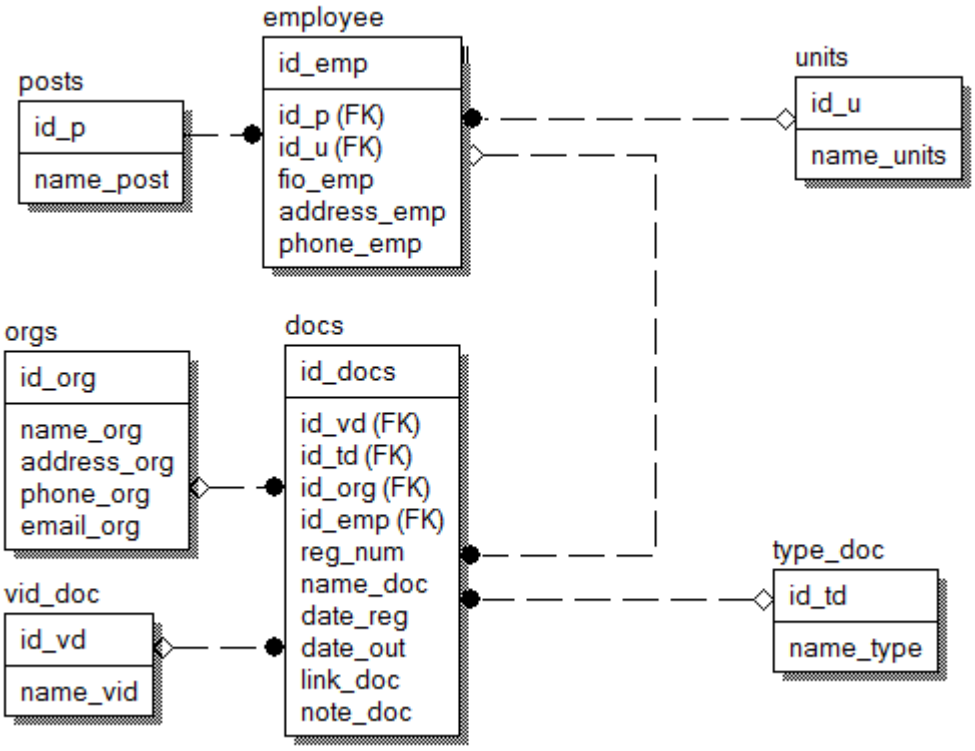

Рис.13. ER-диаграмма ИС «Документооборот казначейского учреждения»

2.2 Проектирование пользовательского интерфейса ИС

2.2.1 Диаграмма прецедентов

Построение диаграмм прецедентов позволит при проектировании информационной системы выявить следующее:

- четко разграничить систему и ее окружение;
- определить, какие действующие лица и как именно взаимодействуют с системой, какой функционал (варианты использования) ожидается от системы;
- определить и описать в словаре предметной области (глоссарии) общие понятия, которые необходимы для детального описания функционала системы (прецедентов)[5].

При построении главной (рис. 14) и дополнительной (рис. 15) диаграммы прецедентов были выявлены и определены основные пользователи ИС и их взаимодействие с ней.

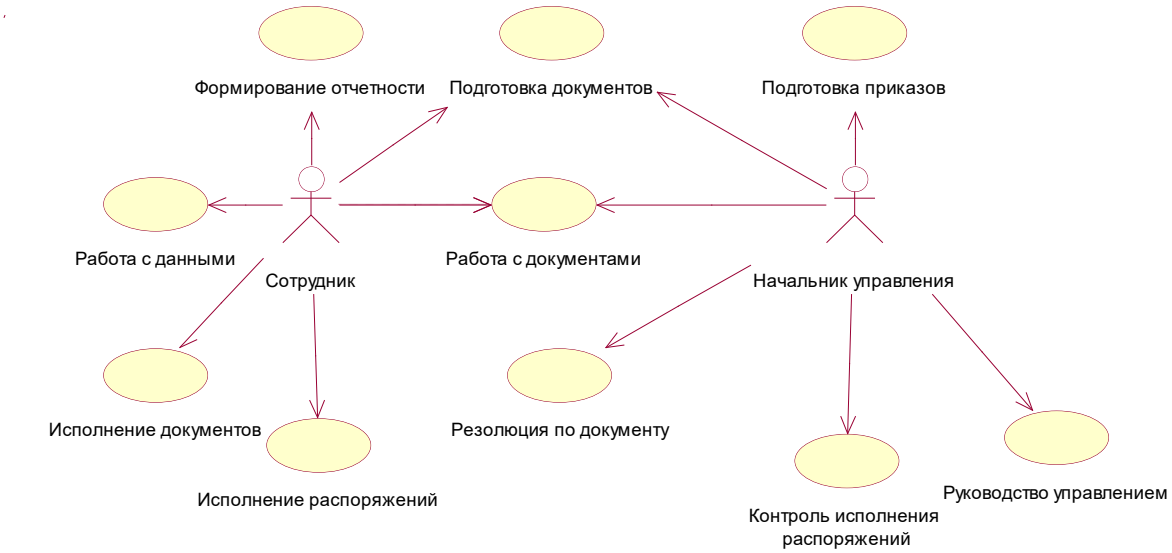

Рисунок 14Главная диаграмма прецедентов ИС

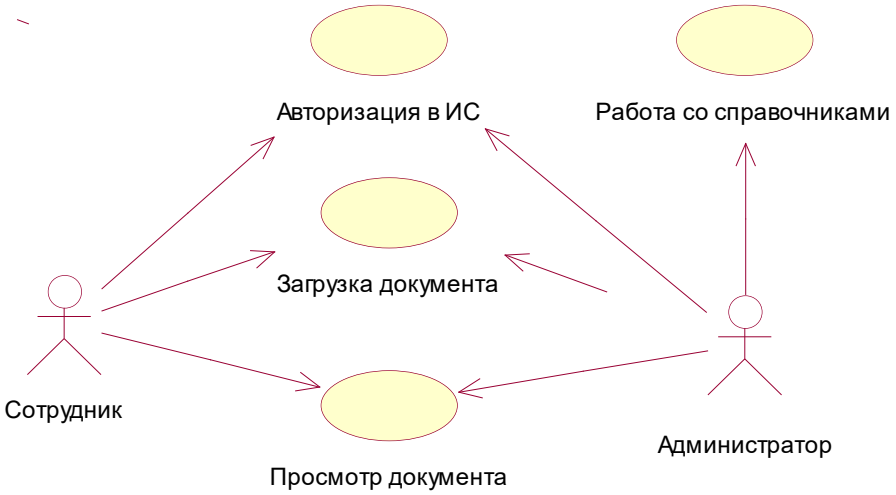

Рисунок 15. Дополнительная диаграмма прецедентов

2.2.2. Эскиз пользовательского интерфейса

Построив диаграмму прецедентов, отражающую основные объекты и принципы взаимодействия с проектируемой информационной системой мы можем перейти к созданию эскиза пользовательского интерфейса (рис. 16).

Рисунок 16. Эскиз пользовательского интерфейса «Документооборота казначейского учреждения

2.2.3. Назначение команд

Информационная система должна позволять пользователям только те действия, которые мы определили при проектировании пользовательского интерфейса.

Пользовательский интерфейс администратора состоит из следующих пунктов:

1. Корреспонденты;
2. Подразделения;
3. Незарегистрированные документы;
4. Зарегистрированные документы;
5. Сотрудники организации.

Пользовательский интерфейс сотрудника состоит из двух команд:

1. Просмотр документов;
2. Регистрация нового документа;
3. Поиск документов по реквизитам.

В таблицах 4 и 5 определены соответствия команд и действий пользовательского интерфейса администратора и сотрудника соответственно.

Таблица 4. Команды интерфейса администратора

Команда	Описание
Корреспонденты	Просмотр и изменение информации об организациях по переписке
Подразделения	Просмотр и изменение информации о подразделениях УФиНП

Незарегистрированные документы	Просмотр и изменение записей о документах
Зарегистрированные документы	Просмотр и изменение данных о зарегистрированных документах
Сотрудники	Просмотр и изменение данных о сотрудниках организации

Таблица 5. Команды пользовательского интерфейса сотрудника

Команда	Описание
Просмотр документов	Просмотр документов
Регистрация документа	Регистрация и заполнение данных о документе
Поиск документов по реквизитам	Поиск документов по определенным реквизитам в ИС

2.3 Реализация ИС «Документооборот казначейского учреждения»

2.3.1 Обоснование выбора для разработки информационной системы

Абстрактное представление данных, сокрытие от пользователя особенностей хранения и управления этими данными – один из основополагающих моментов в проектировании информационной системы. База данных разрабатывается, как общий ресурс для различных пользователей. Каждый пользователь имеет собственное представление о структуре данных, отличное от других пользователей. Это обусловлено следующими причинами:

- каждый пользователь обращается к общим данным, используя при этом своё представление о структуре данных;
- взаимодействие пользователя с базой данных (БД) не должно зависеть от особенностей её физической реализации;
- администратор базы данных (АБД) может изменять структуру и формат данных, без изменения представлений;
- структура БД не должна зависеть от способов физического хранения данных;
- АБД должен иметь возможность изменять концептуальную модель данных и это не должно менять представлений пользователей.

Одним из таких физических изменений может быть изменение способа хранения.

Следовательно, для решения упомянутых задач, помимо выбора модели БД, так же необходимо определиться со способом доступа к БД системы или архитектурной моделью программного комплекса (архитектура БД). Архитектурная модель – модель, с помощью которой СУБД структурирует и манипулирует перманентными данными.

Простейшей архитектурой баз данных является локальная модель. Если БД располагается на том же компьютере, что и приложения, которые работают с этой базой данных, то информационная система имеет локальную архитектуру. С базой данных, как правило, работает один пользователь - это однопользовательский режим.

В локальной архитектуре логика, данные и приложение, работающие как единое целое не могут быть разделены. Среди таких СУБД можно назвать Access, Dbase, Paradox.

Можно организовать многопользовательский режим доступа, если использовать сеть с локальной БД. Файлы БД и приложения располагаются на сервере сети. Для каждого пользователя работает своя копия приложения. Этот вариант использования локальной БД принято считать системой с архитектурой «файл-сервер».

Архитектура «файл-сервер» используется в сетях с небольшим количеством компьютеров. В качестве СУБД могут использоваться персональные СУБД, например, Paradox или dBase. Преимуществом архитектуры «файл-сервер» являются простота построения. Приложение разрабатывается в расчете на одного пользователя.

Если БД поместить на удаленный компьютер-сервер сети, а приложения, размещать на компьютерах пользователей, то такая архитектура называется архитектурой «клиент-сервер». Информационная система будет состоять из сервера и клиента.

Клиентская часть, это приложения пользователей. Клиентская часть формирует и отправляет запрос удаленному серверу с БД. Запрос формируется на языке запросов SQL. Язык запросов к БД SQL является стандартом реляционных моделей данных. Удаленный сервер направляет запрос SQL-серверу баз данных, который обеспечивает выполнение запроса с дальнейшей выдачей результатов клиенту.

При использовании архитектуры «клиент-сервер» клиент получает только данные, которые были затребованы клиентом.

Для построения архитектуры «клиент-сервер» используются многопользовательские СУБД, такие как, Oracle, Microsoft SQL Server, MySQL, PostgreSQL.

Рассмотренная архитектура «клиент-сервер» является двухуровневой: приложение-клиент и сервер БД. Клиентское приложение еще называют «толстым», клиентом (иногда называют «сильным» клиентом). Развитие этой архитектуры ведет к появлению трехуровневой «клиент-серверной» архитектуры: приложение-клиент, сервер приложений и сервер.

В трехуровневой архитектуре часть программного кода приложения-клиента располагается на сервере приложений. Клиентское приложение называют «тонким» клиентом.

Рассмотрев различные архитектуры БД, а так же выявив достоинства и недостатки, каждой из них, было принято решение, что при проектировании разрабатываемого модуля лучше всего воспользоваться клиент-серверной технологией. Данная технология более надежна при многопользовательской работе в отличие от того же сетевого подключения по технологии «файл-сервер».

2.3.2 Разработка ключевых требований к реализации информационной системы

Любая информационная система выполняет ряд стандартных функций:

- сбор информации (автоматизированный или ручной через интерфейс пользователя);
- хранение информации в файлах или в БД;
- обработка информации из БД или внешних источников;
- выдача информации на экран дисплея или бумажные носители.

Программные компоненты информационной системы реализуют эти функции. Хранение информации, это долговременное или кратковременное хранении данных на носителях информации. В настоящее время, как правило, для организации хранения данных и доступа к ним используются СУБД.

База данных состоит из взаимосвязанных данных, используемых несколькими приложениями, пользователями. Данные не зависят от приложений. Для внесения изменений применяется система управления базой данных (СУБД).

Обработка информации заключается в получении одних «информационных носителей» из других «информационных носителей».

Существуют основные базовые процессы обработки информации: проектирование и создание данных; обновление данных; разработка процедур обеспечения целостности данных; поиск информации;

Информационная система (ИС) включает базовые компоненты:

- база данных или совокупность баз данных;
- программные средства обработки данных;
- программные средства поддержки пользовательского интерфейса.

По способам обработки данных ИС делятся на системы с централизованной и децентрализованной организацией данных (это подробно описано в предыдущем разделе).

В централизованных ИС используется одна база данных, обработка данных производятся единственным сервером баз данных. Для децентрализованных ИС база данных состоит из нескольких частей, располагаемых на разных серверах.

Веб-ориентированная ИС - это клиент-серверное приложения, где клиентом является браузер, а сервером — веб-сервер. Программное обеспечение информационной системы расположено на сервере. Клиент-серверная архитектура информационной системы предполагает взаимодействие двух процессов, обменивающихся данными – клиентский процесс и серверный процесс.

Клиентский процесс реализует интерфейс пользователя, транслирует запросы пользователя в запросы к серверу и отправляет эти запросы серверу. Сервер принимает эти запросы, обрабатывает их, и результат обработки передает клиентскому процессу. Возможно обращение сервера приложений

серверу базы данных. Сервер базы данных тоже расположен на серверной стороне. Для веб-ориентированной информационной системы запросы отправляются посредством использования HTTP протокола.

Целью проекта является разработка модуля электронной регистрации и поиска документов, разработка БД, программных составляющих, обеспечивающих администрирование БД, разделение доступа к данным и функциям системы. Поставлена задача разработки БД, веб-интерфейса для двух групп пользователей –сотрудников и администратора ИС, разделение доступа к базе данных и программным компонентам.

С позиции пользователя системы номинальными требованиями являются:

- возможность проведения необходимых операций с исходными данными и результатной информацией (разграничение доступа, ввод, обновление, удаление, поиск);
- наличие понятного и удобного пользовательского интерфейса при работе с базой данных; интерфейса приближенного к привычным журнальным и карточным формам регистрации документов.
- добавление/изменение/удаление групп пользователей – сотрудников, подразделений организации (для администратора системы);
- добавление/изменение/удаление записей регистрации документов и самих документов;
- формирование поисковых запросов по наиболее актуальным атрибутам поиска;
- формирование отчетов за периоды времени и по наиболее актуальным атрибутам поиска (например, договора определенного партнера организации).

Поиск информации и формирование отчетов. ИС должна позволять проводить поиск информации по нескольким ключам поиска информации:

- поиск документов по номеру – позволяет осуществлять поиск документов по номеру;
- поиск документов по номеру и дополнительным атрибутам позволяет осуществлять поиск документов по ряду параметров;
- поиск документов по наименованию – позволяет осуществлять поиск документов по наименованию;
- поиск документов по дате регистрации – позволяет осуществлять поиск документов по дате регистрации;
- поиск сотрудника организации по должности – позволяет осуществлять поиск сотрудника по должности (например, сотрудника, который регистрировал документ);
- поиск сотрудника организации по фамилии – позволяет осуществлять поиск сотрудника по фамилии;
- отчеты по различным видам документов за определенные периоды времени;

Возможные атрибуты поиска могут уточняться на этапе опытной эксплуатации системы.

Рассмотрим общие требования к системе в целом и ее программным компонентам.

Экранные формы. ИС должна обеспечить ввод и модификацию данных посредством экранных форм, представленным в виде таблиц. Форма предполагает наличие полей для ввода и модификации данных. Программное обеспечение должно быть независимым от структуры таблиц базы данных.

*Интерфейс пользователей***.** ИС должна предоставлять интуитивно-понятный интерфейс пользователей, обеспечивающий простоту работы на всех ее этапах.

*Защита от несанкционированного доступа***.** Вход в защищенные области системы, БД, поисковые компоненты должен осуществляться после ввода пользователем логина и пароля. Для этого разрабатывается или используется подсистема аутентификации пользователя. После ввода логина и пароля, система определяет статус и полномочия пользователя по доступу к данным и функциям системы.

Сохранность информации. Для сохранности информации администратор системы производит операции по созданию дампов базы данных (резервных копий). Частота создания резервной копии определяется на этапе опытной эксплуатации системы.

Требования к базовым программным средствам.

Клиентская часть информационной системы работает в среде Windows.

Программное обеспечение серверной части ИС реализуется в ОС Linux.

Языком программирования серверной части ИС является PHP.

Разрабатываемая ИС предполагает хранение большого объема информации, должна обеспечивать анализ и оценку данных.

Указанным требованиям отвечают системы управления базами данных (СУБД), одна из которых и должна быть выбрана для практической реализации данного дипломного проекта. Современные СУБД обладают следующими свойствами:

- представляют собой удобный и современный, предназначенный для разработки и эксплуатации информационных систем, предоставляя

интегрированную среду проектирования, работа в которой базируется на манипулировании с объектами и их атрибутами;

- позволяют для обработки информации и быстрого формирования решений привлекать все преимущества реляционных баз данных.

С позиций распространенности и доступности рекомендуется использование при программировании среды СУБД MySQL.

2.3.3 Ожидаемые риски и их описание

Возможны следующие виды рисков и причины их вызывающие.

Организационные риски:

– Отсутствие или некорректное формирование целей или задач проекта приведет к непониманию целесообразности разработки

– Недостаточное изучение объекта автоматизации влечет за собой неправильное формирование требований и функций системы

Технологические риски:

– Недостаточное обследование компании может привести к разработке неполного функционала, что приведет к неверному ожиданию заявленного функционала на стадии эксплуатации

Субъективные риски:

– Сопротивление со стороны сотрудников или руководителей компании к внедрению системы, отсутствие желания обучаться новому программному продукту и методам работы

Данные риски могут затянуть разработку системы на неопределённый срок, или серьезно повлиять на ожидаемый конечный результат со стороны компании.

Для минимизации или полному исключению данных рисков, следует применить следующие аспекты:

- На стадии проектирования вести активные переговоры, как с руководителями, так и с рядовыми сотрудниками компании, чтобы учесть все пожелания для обеспечения более благоприятных условий работы в новой системе;
- Применение типовых решений там, где это возможно, для ускорения процесса разработки;

2.3.4 Проектирование структуры БД

На основании анализа исходных данных проведенных при проектировании ИС были выделены следующие сущности концептуальной модели данных для регистрации входящих документов организации.

1. *«Корреспонденты»*.Отправители документов – организации, условно названы клиентами системы.
2. *«Подразделения организации»*.Сущность, содержащая данные о подразделениях организации;
3. *«Документы»*. Сущность, содержащая данные о входящих документах;
4. *«Статус документа»*. Сущность, содержащая данные о регистрации документа сотрудником организации;
5. *«Сотрудники»*. Сущность, содержащая данные о сотрудниках организации;

При разработке базы данных объекты преобразуются в таблицы БД, а атрибуты объекта преобразуются в поля таблиц БД (Приложение А).

Структура БД может меняться в процессе разработки ИС, но при этом все заголовки таблиц на русском языке, которые находятся в специальных блоках

программных модулей, также должны соответственно меняться. Это не влияет на разработку самих программных модулей, реализующих табличное представление данных на экране дисплея.

Для проектирования БД и построения ER- диаграммы использовался пакет *MySQLWorkbench*, который является инструментом проектирование БД и их обслуживания. Пакет автоматизирует, улучшает связь среди команд разработчиков и архитекторами БД. Он позволяет архитекторам данных наглядно представлять требования, общаться с заинтересованными лицами по вопросам проектирования. Это свойство пакета использовалось для построения ER- диаграммы БД системы.

Большинство веб-приложений базируется на серверах общего доступа. Доступ к таким хостам осуществляется посредством FTP- программ типа Filezilla, которая использовалась при разработке системы.

Для дополнительных манипуляций с базой данных на хостинге был использован пакет phpMyAdmin. Эта программа является web-приложением, представляющее собой визуальный табличный интерфейс для работы с базами данных. Этот пакет обеспечивает средства визуального создания базы данных, составляющих таблиц и генерацию на лету SQL.

2.3.5 Описание работы модулей системы

Для разработки программного обеспечения настоящего приложения был проведен системный анализ и выбраны базовые функции, подлежащие реализации. На основе этого проведен выбор средств языков программирования, реализующих базовые функции.

Разработка программных модулей не велась с нуля, а были существенно модифицированы необходимые функции и классы PHP, взятые из книг. Несмотря на это процент новых доработок адаптированных под настоящее приложение составляет около 90%.

Для соединения с базой данных был использован класс**,** реализующий основные запросы к базе данных. Еще одной важной базовой компонентой программного обеспечения является модифицированный класс таблиц, измененный и дополненный новыми методами и функциями. Данный класс использовался как базовый для создания представлений всех таблиц в едином виде, независимо от содержания таблиц.

Главная страница веб-приложения расположена по адресу: http://demo36647.atservers.net/index.php . Для доступа в закрытые части системы используются следующие учетные данные. Для администратора ИС логин=admin, пароль=cthdth, для пользователя – логин=nimda, пароль=ljrevtyn.

Структурно эта веб-страница состоит из ряда, специальным образом построенных таблиц в терминологии языка HTML, разделяющих экран на отдельные поля и линейного, в виде ленты, меню с возможными вариантами работы с системой (рис. 17). Каждой кнопке этого меню соответствует ссылка на определенный раздел ИС. Нажимая на заголовки, пользователь инициирует выполнение перехода в соответствующий раздел.

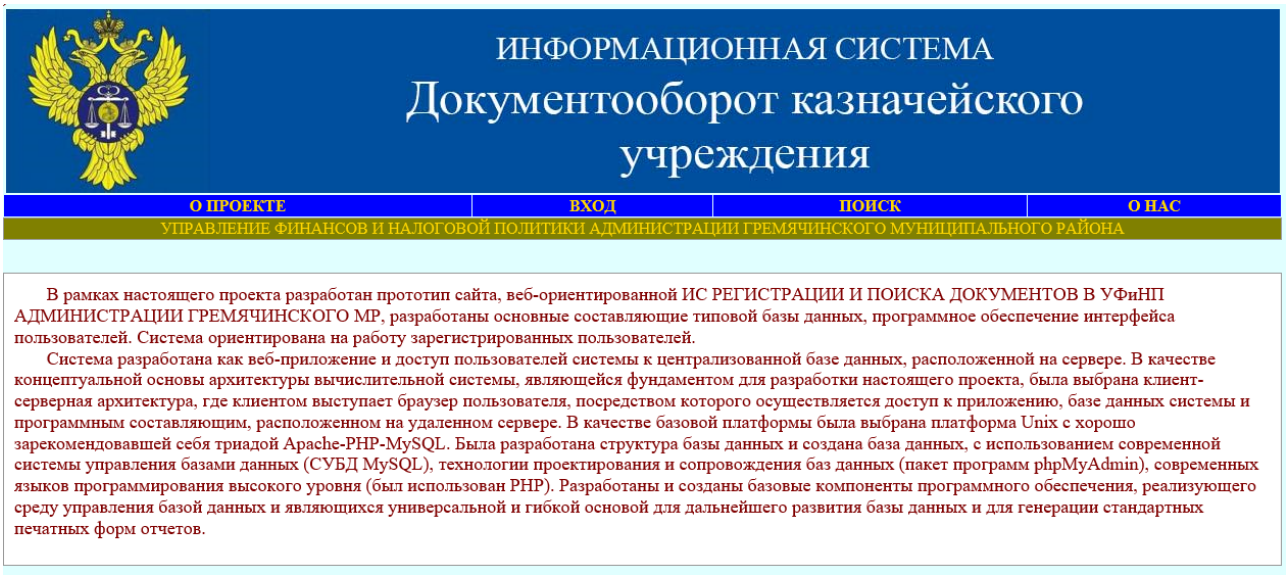

Рисунок 17. Главное окно ИС «Документооборот казначейского учреждения»

Для поиска документов переходим на вкладку главного меню «ПОИСК»(рис 18). Если в первом поле зададим наименование документа «дог» (имеется в виду «договор»), а в поле года регистрации внесем«2016-01» (первый месяц 2016 года), то получим все договора, зарегистрированные в первом месяце текущего года (рис. 19).

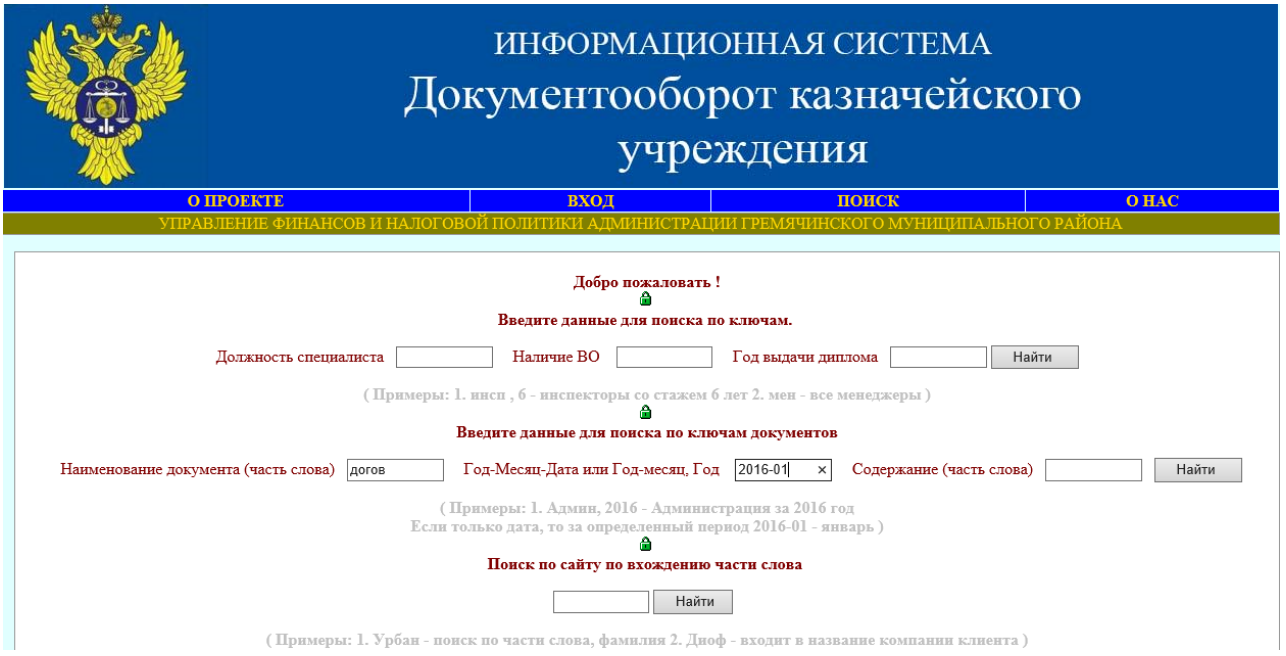

Рисунок 18. Ввод параметров для поиска договоров за первый месяц текущего года

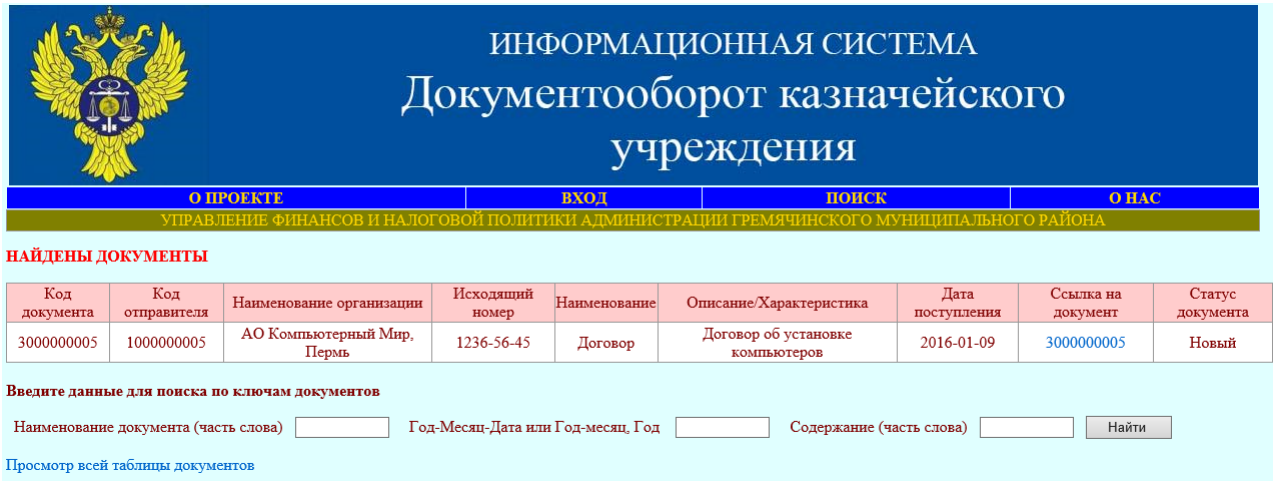

Рисунок 19 – Результат поиска договоров за первый месяц текущего года

Можно задать название компании, тогда будут найдены все входящие документы, пришедшие от этой компании. Таким образом, можно задавать различные комбинации ключевых параметров поиска. Обычно наиболее оптимальные параметры определяются в процессе опытной эксплуатации системы.

Выводы

В заключении второй главы можно подвести следующие итоги. В процессе разработки ИС «Документооборот казначейского учреждения» был произведен комплексный анализ выявления необходимых требований для реализации поставленных задач, а именно:

- Создание концептуальной модели ИС;
- Разработка пользовательского интерфейса;
- Произведён выбор СУБД и языка программирования высокого уровня для реализации ИС;

Следует отметить, что без этих предварительных шагов реализация серверной части информационной системы представляла собой не подъемную задачу. Только после вышеуказанных манипуляций с информацией, поступившей от заказчика, стала возможной реализация практической части разработки ИС.

3. Определение эффективности внедрения ИС «Документооборот казначейского учреждения»

Любая деятельность человека имеет целесообразность и смысл, насколько эффективной она является. Эффективность любых действий можно определить экономичностью хозяйствования и получением оценки этих результатов, которые удалось получить в производстве. Таким образом, определяется экономический эффект любой компании для того, чтобы понять, насколько такое ведение бизнеса является актуальным для его владельца.

Таким образом, экономический эффект производства определяется как оптимальное применение ресурсов в соответствии с общественными потребностями. Отметим основные критерии экономической эффективности какого-либо предприятия:

- затраты живого труда на производство продукции или оказание услуг;
- затраты материальных ресурсов;
- затраты финансовых ресурсов;
- выход продукции или услуг.

Следует так же учесть, что успешность предприятия зависит не только от экономических показателей, но и социальной составляющей деятельности организации.

Социальная эффективность отражает результат управленческой деятельности степенью использования потенциальных возможностей коллектива для осуществления миссии фирмы и ее общественной значимости.

Проведение экономического исследования по внедрению ИС «Документооборот казначейского учреждения» состоит в определении экономического эффекта от внедрения программного продукта.

Цель проводимого исследования заключается в определение сроков окупаемости внедряемой системы.

3.1 Расчет затрат на разработку ИС «Документооборот казначейского учреждения»

3.1.1 Себестоимость программы

Рассчитать себестоимость любого программного продукта или услуги - это определение совокупности затрат, которые можно разделить на следующие позиции:

1. Заработная плата основной деятельности;
2. Накладные расходы.

Расчет первой категории производится в соответствии с трудоемкостью программного продукта.

Фонд времени работ по плану – 22 дня при 8-ми часовом рабочем дне, что в свою очередь определяет общее количество рабочих часов в месяц – 176. На разработку информационной системы затрачено 31 день. Из чего следует, что объем затраченного времени составил 248 часов.

Размер средней заработной платы PHP-программиста начального уровня в Пермском крае составляет 24000 рублей.

Начнем с определения часовой тарифной сетки ($С_ч$) по формуле (1):

$$С_ч = \frac{Оклад}{Ф_{РВ}}, (1)$$

где $Ф_{РВ}$ – плановый фонд рабочего времени за месяц.

$С_ч$= 136,4 руб. в час

Основную заработную плату следует рассчитать по следующей формуле (2):

$$ЗП_{осн} = С_ч * Т_{ожс}=136,4 * 248, (2)$$

где $Т_{ожс}$ - затраченное время на создание ИС, которое составило 31 день (248 часов).

$$ЗП_{осн} = 136,4 * 248 = 33827,2 руб.$$

В расходы включены затраты на электроэнергию, потребляемую компьютером за время разработки программы и амортизационные отчисления.

Таблица 8. Затраты на электроэнергию

Вид оборудования	Кол-во (шт.)	Мощность, кВт	Стоимость 1 кВт/час, руб.	Время работы оборудования, час	Сумма затрат, руб.
Компьютер	1	0,15	3,37	248	125,36
Итого	-	-	-	-	125,36

В таблице 8 выполнены расчеты по затрате ресурсов на электроэнергию по формуле (3):

$$Сумма = (М * С) * Т = 0,15*3,37*248 = 125,36, \quad (3)$$

где:

- М - мощность, кВт;
- С - стоимость 1 кВт/час, руб.;
- Т - время работы оборудования, час.

3.1.2 Амортизационные отчисления

Оборудование имеется в наличии, соответственно рассчитаем годовую сумму амортизации.

Таблица 9. Амортизация оборудования

Вид оборудования	Первоначальная стоимость, руб.	Количество рабочих часов	Норма амортизации, %	Сумма отчислений по амортизации, руб.
Компьютер	22800	248	12,5	357,69
Итого	-	-	-	357,69

В таблице 9 рассчитана сумма амортизационных отчислений за период разработки программного продукта в часах.

Амортизация оборудования за период работы над созданием ИС определяем по формулам (4) и (5):

$$А_{г} = \frac{С_{п} \cdot Н_{а}}{100} = \frac{22800 \cdot 12,5}{100} = 2850, \quad (4)$$

$$А_{факт} = \frac{А_{г} \cdot Т_{факт}}{N} = \frac{2850 \cdot 248}{1976} = 357,69, \quad (5)$$

где:

- $А_{г}$ – сумма отчислений по амортизации в год, руб.;
- $С_{п}$ – стоимость оборудования на начало эксплуатации, руб.;
- $Н_{а}$ – годовая норма амортизации, %;
- $А_{факт}$ – амортизация за период разработки программы, руб.;
- $Т_{факт}$ – количество затраченных часов задействованного оборудования (44 часа);
- N – годовой объем рабочих часов (247 дней * 8 часов = 1976 часов).

Накладные расходы суммируются из затрат на электроэнергию и отчисления по амортизации.

Сумма накладных расходов = 357,69 + 125,36 = 483,05 руб.

3.1.3 Эксплуатационные материалы

За счет использования бесплатных и trial-версий программных продуктов удалось снизить стоимость эксплуатационных материалов до нуля.

Таблица 10.Эксплуатационные материалы

№	**Наименование**	**Стоимость 1 ед., руб.**	**Сумма, руб.**
1	Open Server	0руб	0руб
2	Rational Rose	0 руб.	0 руб.

3	Erwin Process Modeler	0 руб.	0 руб.
4	Erwin Data Modeler	0 руб.	0 руб.
5	NetBeans	0 руб	0 руб
ИТОГО:			0руб

В таблице 11 отражены фактические расходы на разработку ИС.

Таблица 11.Калькуляция затрат на разработку программного продукта

Статья затрат	Сумма затрат, руб.
Общая сумма з/п	33827,2
Накладные расходы, в т.ч. амортизация	483,05
Итого:	34310,25

Следовательно, затраты на разработку программного обеспечения составили 34310,25 рублей.

3.1.4 Определение возможной цены программного продукта

Рассчитаем плановый уровень прибыли с условием, что процент рентабельности составляет 25%. Прибыль рассчитывается по формуле (6):

$$П = \frac{С_{полн} \cdot Р}{100} = \frac{34310,25*25}{100} = 8577,56, \quad (6)$$

где:

1. $С_{полн}$ – себестоимость;
2. Р - процент рентабельности.

Цена программного продукта соответствует сумме полной себестоимости и прибыли и рассчитывается по формуле (7):

$$Ц = С_{полн} + П = 34310,25 + 8577,56 = 42887,81 \text{ руб.} \quad (7)$$

НДС и цена программного продукта с НДС определяется по формулам (8) и (9):

$$НДС = \frac{Ц \cdot 18}{100} = \frac{42887,81*18}{100} = 7719,81 \quad (8)$$

Цена = НДС + Ц= 7719,81+ 42887,81 = 50607,62 (9)

Таким образом, предполагаемая стоимость программного продукта составляет 50607,62 рублей.

3.1.4 Расчет основных затрат

К основным затратам относятся приобретение информационной системы и обучение персонала. Совокупность основных затрат на внедрение ИС «Документооборот казначейского учреждения» приведена в таблице 12.

Таблица 12.Затраты на внедрение ИС

№	Статья затрат	Стоимость за единицу	Количество	Сумма, руб.
1	Приобретение программы	50607,62 руб./шт.	1 шт.	50607,62
2	Обучение администраторов	300 руб./ч.	3 ч.	900
	Итого:	–	–	51507,62

Основной статьей расходов на внедрение является приобретение программного продукта. Затраты составляют 51507,62 рубля.

Обслуживанием программного продукта занимаются администраторы, которые входят в штат сотрудников школы, поэтому затраты ограничиваются обучением сотрудников администрированию продукта. Стоимость обучения администраторов составляет 900 руб.

Стоимость внедрения ИС «Документооборот казначейского учреждения» составляет 51507,62 руб.

3.2 Эффективность внедрения ИС

Внедрение информационной системы «Документооборот казначейского учреждения» в УФиНП позволит улучшить производственные показатели, которые делятся на 2 группы:

1. Экономический эффект:
    - Уменьшение ассигнования средств на расходные материалы для офисной техники;
    - Сокращение трудозатрат сотрудников.
2. Социальный эффект:
    - Создание единого информационного пространства;
    - Прозрачность управления;
    - Контроль исполнительской дисциплины;

Прямая экономия средств заключается в сокращении времени поиска документов сотрудниками УФиНП и затрат на расходные материалы для печати документов.

В месяц организация закупает для ведения документооборота:

1. Бумага – 3 упаковки.

   1 упаковка бумаги стоит 210 руб., средний месячный расход на покупку бумаги составляет 3 * 210 = 630 руб./мес.
2. Средняя стоимость совместимого картриджа для монохромного лазерного принтера – 1100 руб.

Количество принтеров в учреждении, нуждающихся в заправке картриджей для печати корреспонденции – 2 шт./мес. Затраты на приобретение картриджей составляют 2*1100 = 2200 руб./мес. Итого на расходные материалы в месяц используется: 630+2200 = 2830 руб.

3.2.1 Оценка сокращения временных трудозатрат

Сотрудник тратит на поиск необходимых документов ориентировочно 10% рабочего времени.

При использовании ИС «Документооборот казначейского университета»:

- Сокращение времени на поиск необходимой информации;
- Отсутствие ожидания готовой отчетности;
- Уменьшение времени на формирование отчетности;

Средний процент экономии времени на оформление и передачу отчетности, а также на поиск необходимой информации – 45%. Соответственно, использование ИС «Документооборот казначейского учреждения» позволит уменьшить расход рабочего времени на (10% * 45%)/100% = 4,5 %.

3.2.2 Оценка ежемесячных затрат на одного сотрудника, использующего ИС «Документооборот казначейского учреждения».

Работой с документами и поручениями у нас занят каждый сотрудник, в том числе делопроизводитель. Уровень средней заработной платы в организации примерно равен 16 тысячам рублей.

Произведем расчет ежемесячных затрат на одного сотрудника, работающего с ИС «Документооборот казначейского учреждения»: 16000*0,045=720руб. В связи с тем, что с системой будут взаимодействовать 12 человек, ежемесячный экономический эффект составит 720*12=8640 руб.

Далее суммируем стоимость экономии на офисных расходных материалах (2830 руб.) и на рабочее время (8640 руб.) получим, что внедрение ИС влечет за собой годовой экономический эффект в (2830+8640)*12=137640руб.

3.2.3 Срок окупаемости

Рассчитаем срок окупаемости продукта по формуле (10):

$$Т= К / П, \quad (10)$$

,где

- К – единовременные капитальные затраты при внедрении;
- П – годовая экономия, руб.

В нашем случае К = 51507,62руб., П = 137640руб.

Подставляя данные значения в формулу (10), получим: Т = 51507,62/ 137640= 0,37. Умножим на 365 дней в году и получим срок окупаемости продукта в 135,5 дня.

Следовательно, внедрение ИС «Документооборот казначейского учреждения» превысит уровень окупаемости через 135,5 дня.

Выводы

Исходя из произведенных расчетов экономических показателей, можно сделать следующие выводы, что разработка и внедрение ИС позволит нам:

1. снизить трудоемкость операций с документами;
2. устранить издержки на покупку расходных материалов;
3. повысить социальный эффект управления организацией.

Заключение

Объектом исследования дипломного проекта является деятельность служб «Управление финансов и налоговой политики администрации Гремячинского муниципального района», осуществляющих регистрацию и поиск документов.

Предметом исследования является автоматизация процессов регистрации и поиска документов, поступивших и созданных в организации.

В рамках дипломного проекта поставлена цель: повышение эффективности деятельности служб организации по учету документооборота посредством внедрения модуля электронной регистрации и поиска входящих документов.

Для достижения поставленной цели были решены следующие задачи:

- анализ деятельности служб организации по регистрации и поиску документов;
- анализ документооборота организации;
- экономическое обоснование системы;
- разработка концептуальной модели предметной области;
- разработка интерфейсных решений;
- разработка обобщенного алгоритма работы системы;
- разработка базы данных и программных средств электронной регистрации и поиска документов.

На основании проведенного анализа деятельности конкретной организации сделаны выводы и предложена типовая архитектура программного обеспечения и концептуальная модель базы данных, которые могут быть применимы, настраиваемы для любой организации такого типа.

В качестве одной из возможных концепций создания информационных систем документооборота и модуля регистрации и поиска входящих

документов, рассматривается клиент-серверная архитектура, а в качестве операционной платформы для реализации предлагается платформа Linux.

Разработано программное обеспечение ИС для электронной регистрации и поиска входящих документов на языке PHP с использованием СУБД MySQL, реализующее методы регистрации и поиска входящих документов по ряду поисковых параметров.

Внедрение системы обеспечит:

- прозрачность движения документов;
- гарантированное наличие информации и моментальный доступ к ней;
- эффективный поиск информации по документам, формирование статистических данных;
- ускоренный обмен документами и сроков исполнения в виду отсутствия необходимости в физическом размножении и перемещении документов;
- управляемые и контролируемые информационные потоки и как следствие повышение управляемости организацией в целом.

Библиографический список

1. «Википедия». Документ https://ru.wikipedia.org/wiki/Документ (дата обращения: 01.05.2016)
2. «Организация работы с документами»http://management-rus.ru/economics/orgrab.php (дата обращения: 03.05.2016)
3. Электронный документооборот: что такое электронный документооборот, основные понятия, виды, преимущества, задачи, критерии выбора, классификация систем, требования http://www.docflow.ru/edu/glossary/detail.php?ID=27946 (дата обращения: 04.05.2016)
4. Журнал «Управление персоналом: hr, рекрутмент, маркетинг, интервью с топ менеджерами и бизнес-элитой». Развитие принципов документооборота при переходе от бумажного к электронному взаимодействию. М. П. Бобылева, к. и .н., доцент http://www.top-personal.ru/officeworkissue.html?224 (дата обращения: 06.05.2016)
5. «ИНТУИТ». Введение в UML http://www.intuit.ru/studies/courses/1007/229/lecture/5962?page=1 (дата обращения: 13.05.2016)
6. Документация PHP. http://php.net/manual/ru/index.php (дата обращения 14.05.2016)
7. Изучение PHP. http://phpfaq.ru/ (дата обращения 14.05.2016)
8. Справочник языка PHP. https://php.ru/manual/ (дата обращения 15.05.2016)
9. PHP, MySQL и другие вэб-технологии. www.php.su (дата обращения 15.05.2016)

10. SQL.ru – всё про SQL, базы данных, программирование и разработку информационных систем. http://www.sql.ru/ (дата обращения 16.05.2016)

11. *Д.Н. Колисниченко* PHP и MySQL. Разработка Web-приложений 5-е издание / Д.Н. Колисниченко. – С.Петербург, 2015.

12. *Ларри Ульман* PHP и MySQL. Создание интернет-магазинов. 2-е издание / Ларри Ульман. – Вильямс, 2015.

13. *Р. Никсон* Создаем динамические веб-сайты с помощью PHP, MySQL, javascript, CSS и HTML5. 3-е изд / Р. Никсон. – С.Петербург, 2015.

14. *Кевин Янк* PHP и MySQL. От новичка к профессионалу / Кевин Янк. – М, 2015.

15. *Е.А. Лопатникова* Делопроизводство. Образцы документов с комментариями / Е.А. Лопатникова. – М, 2009.

16. Энциклопедия делопроизводства. http://www.edou.ru/enc/ (дата обращения 20.05.2016)

17. Делопроизводство. http://www.bibliotekar.ru/deloproizvodstvo-1/index.htm (дата доступа 25.05.2016)

18. *Е.П. Смирнова* Делопроизводство для секретаря / Е.П. Смирнова. – М, 2010.

19. *Бобылева М.П.* Управленческий документооборот. От бумажного к электронному / Бобылева М.П. – Самиздат, 2010.

20. **Российская Федерация. Законы.** Постановление Правительства РФ от 15 июня 2009 г. N 477 "Об утверждении Правил делопроизводства в федеральных органах исполнительной власти" (с изменениями и дополнениями). – Гарант, 11с.

Приложение А.

Таблица 13. Организации

Объект	Таблица БД	Атрибуты	Содержание	Первичный ключ
Организация	orgs	id_org name_org address_org phone_org email_org	Идентификатор Наименование Адрес Телефон Адрес электронной почты	id_org

Таблица 14. Подразделения

Объект	Таблица БД	Атрибуты	Содержание	Первичный ключ
Подразделения	units	id_u name_units	Код подразделения Наименование	id_u

Таблица 15. Документы

Объект	Таблица БД	Атрибуты	Содержание	Первичный ключ
Документы	docs	id_docs id_vd	Код документа Вид документа	id_docs

		id_td id_org id_emp reg_num name_doc date_reg date_out link_doc note_doc	Тип документа Код организации Код сотрудника Регистрационный номер Наименование Дата регистрации Дата отправки Ссылка на документ Аннотация	

Таблица 16. Должность

Объект	Таблица БД	Атрибуты	Содержание	Первичный ключ
Должность	posts	id_p name_posts	Код должности Наименование	id_p

Таблица 17. Сотрудники

Объект	Таблица БД	Атрибуты	Содержание	Первичный ключ
Сотрудники	employee	id_emp id_p id_u fio_emp address_emp phone_emp	Код сотрудника Код должности Код подразделения Ф.И.О. Адрес Телефон	id_emp

Приложение Б

```
<?php
require_once 'Header.php';
?>
<p></p>
<p></p>
<p></p>
<table bgcolor="#999999" height=20 width="99%" cellspacing="1" cellpadding="10" border="0">
</table>
<p></p>
<p></p>
<p></p>
<p></p>
<p></p>
<table bgcolor="#999999" height=20 width="99%" cellspacing="1" cellpadding="10" border="0">
<tr>
<td align="left" valign="top" bgcolor="#FFFFFF">

<font  color="Maroon" size="4">
```

<h4>О ДОКУМЕНТООБОРОТЕ В УФиНП АДМИНИСТРАЦИИ ГРЕМЯЧИНСКОГО МР</h4>

В «Управлении финансов и налоговой политики администрации Гремячинского муниципального района» документооборот построен следующим образом: вся входящая, исходящая и внутренняя корреспонденция регистрируется делопроизводителем на бумажном носителе, представленном в виде журналов.

Всего журналов для учета поступившей, отправленной и внутренней корреспонденции три:

1. Переписка с разными министерствами;

2. Переписка с Министерством финансов;

3. Переписка с разными организациями и учреждениями.

Доведение входящей корреспонденции до исполнителя производится путем печати документа и его вручения на обработку.

Отправление документов после исполнения выполняется путем его регистрации в журнале и отправкой посредством электронной или обычной почты,

в зависимости от вида корреспонденции. Внутренние документы, как правило, не регистрируются, а подшиваются в личное дело сотрудника,

в отношении которого были созданы.

Существует необходимость в автоматизации данного процесса при помощи современных подходов к информационным потокам в УФиНП,

а именно создании информационной среды для решения основных задач оборота документов.

В сложившейся ситуации руководителем УФиНП было принято решение по автоматизации данного процесса, путем создания своей ИС документооборота.

</td>

</tr>

</table>

<?php require_once 'Footer.php'; ?>

```
<?php
#####################################################################
################
# Модуль выводит на экран содержимое базы данных 'ufinp' по-таблично.
          #
# Полностью все данные не выводятся на экран, а только первые строки,чтобы
показать#
# возможные действия с таблицами базы данных.                          #
#####################################################################
################
#
require_once            'DB_Tables_Fields.php';  //!!!//
$Arr_acc              =  array();
$Row                  =  array();
$dbtable              =  $FormTypes;
//$head_color   = 'silver';
# Класс для построения таблиц по заголовкам и содержанию таблиц
require_once 'Any_Table_HTML_New.php';

# Класс для работы с базой данных MySQL, описание в теле модуля
# require_once 'DB_Class.php';
# Объект класса работы с базой данных создается внутри модуля DB_Class.php
```

```
# $dbObj = new MySQL_DB(DB_HOST, DB_USER, DB_PWD, DB_NAME);

###

# Так мы делаем софт более независимым от конкретных таблиц БД

#

# Заголовки всех таблиц веб проекта, которые появляются на экране

include_once 'Table_Headers.php'; //!!!// если уже созданы русские заголовки

#

# Заголовки всех таблиц БД проекта, на английском

# require_once 'Table_Headers_DB.php'; //!!! В этой версии загружаются из БД

# // на лету рограммой DB_Tables_Fields.php

require_once 'Header.php'; // заголовок для страниц проекта

require_once 'FormMsgs.php'; // все сообщения программ проекта

#

#

$f=0;  // номер элемента $FormType

foreach ($dbtable as $value) {

#

$headers = $Table_Headers [$FormTypes[$f]];// наименования столбцов таблице
```

```
$headers_db = $Table_Headers_DB[$FormTypes[$f]];// наименования доменов в таблице
#                                    // БД и <input name's в формах
###
# Создаем объекты класса для вывода таблиц, вызываем конструктор
# $handler     = $FormType.'_Handler'; // обработчик настоящей формы

$Table_view =  new HTML_Table($headers , $heard_color);
#
#запрос на выдачу всей таблицы

$QueryResult = $dbObj->query("SELECT * FROM $dbtable[$f] LIMIT 0,2");
# возвращает массив ассоциативных массивов (см. класс DB_Class.php)
# по сути двумерный массив, поэтому используем конструкцию foreach
# это можно увидеть убрав знак # перед print см. ниже. Таким образом
# формируются массивы для вывода на экран для каждой таблицы
#
if ($QueryResult==false)
   {
    $msg = 'Нет записей в базе данных!';
   }
```

```
else
  {
   $NA  = count($QueryResult);
   //print '$NA='.$NA;
   //print '<pre>'; print_r($QueryResult); print '</pre>';
  }
foreach ($QueryResult as $row    => $Arr_acc)
     {
      $i=0;
      foreach ($headers_db as $element)
            {
           $Row          =          array_merge($Row,          array($headers[$i]          =>
$Arr_acc[$headers_db[$i]]) );
          $i++;
           }
      $Table_view   ->  AddRowAssArr($Row);
       }
print                  "<p></p><font                  color=                  \"blue\"
><strong>$All_Tables_Names[$f]</strong></font>"
    ."$Comment_to_Table_up[$f]";
      $Table_view      ->  PrintArr();
print "<a href=\"Tables_Handler.php?FormType=$FormTypes[$f]\">
```

```
      Все данные Таблицы \"$All_Tables_Names[$f]\"</a>"
    .'<p></p>';
//print "<a href=\"$FormType[$f]_Handler.php?FormType=$FormType[$f]\">
//      Все данные Таблицы \"$All_Tables_Names[$f]\"</a>"
#
$f++;                          // новое значение $FormType, новая таблица
}
#
include 'Footer.php';          // стандартная нижняя строка каждой страницы
#
?>
```

```
?php
class Any_Table
        {
         var $table_array = array();
         var $headers     = array();
         var $domns;
         function  Any_Table ($headers)
                        {
                         $this->headers = $headers;
                         $this->domns   = count($headers);
                        }

         function AddRow ($row)
                        {
                         If  ( count ($row) != $this->domns)
                             return false;
                         array_push( $this -> table_array,  $row);
                         return true;
                        }
         function SetArr($Arr)
                             {
```

```
            $this -> table_array = $Arr;
            return true;
           }
function  AddRowAssArr ($row_ass)
          {
           $row=array();
           foreach ($this -> headers as $header)
           { if (!isset($row_ass[$header]))
                $row_ass[$header]='';
             $row[ ]=$row_ass[$header];
           }
           array_push( $this -> table_array,  $row);
           return true;
          }
function PrintArr ()
          {
            print "<pre>";
            foreach ($this -> headers as $header)
                  print '  '."<strong>$header</strong>".' ';
            print "\n";
            foreach ($this -> table_array as $string)
```

```
                    {
                    foreach ($string as $elemnt)
                            print '   '."$elemnt".'   ';
                    print "\n";
                    }
                }
        }
class HTML_Table extends Any_Table
    {
        var $bgcolor;
        var $bgcolor_h = "#FFCCCC";
        var $bgcolor_c = "#FFFFFF";
        var $cellpadding = "1";
        function HTML_Table($headers, $bg='#ffffff' )
            {
                Any_Table::Any_Table($headers);
             $this->bgcolor=$bg;
            }
        function SetCellPadding ($padding)
            {
                $this -> cellpadding = $padding;
```

```
        }
    function PrintArr()
        {
            print "<table bgcolor=\"#999999\" cellspacing=\"1\" cellpadding=\"$this->cellpadding\" border=0>";
            print "<tr>";
            foreach ($this -> headers as $header)
          print "<td align=\"center\" bgcolor=\"$this->bgcolor_h\">$header</td>";
        print "</tr>";
        foreach ($this -> table_array as $row=>$cells)
            {
             print "<tr>";
            foreach ($cells as $elemnt)
                print "<td align='center' bgcolor=\"$this->bgcolor_c\" >$elemnt</td>";
            print "</tr>";
             }
        print '</table>';
            }
  function OutForm($handler)
        {
```

```
$width_h = array(170, 360);
    print "<form name='uniform' action=\"$handler\" method=post>";
    print "<table width=\"99%\" bgcolor=\"#999999\" cellspacing=\"1\" cellpadding=\"$this->cellpadding\" border=0>";
    print "<tr>";
    $i=0;
        foreach ($this -> headers as $header){
            print "<td width='".$width_h[$i]."' align=\"left\" bgcolor=\"$this->bgcolor_h\"> $header</td>";
            $i++;
            }
print "</tr>";
$i=0;
foreach ($this -> table_array as $row=>$cells)
    {
     print "<tr>";
    foreach ($cells as $elemnt)
        {
         print "<td width='".$width_h[$i]."' align='left' bgcolor=\"$this->bgcolor_c\" >$elemnt</td>";
         if ($i==0) $i++; else $i--;
        }
```

```
                print "</tr>";
                 }
            print '</table>';
            print '</form>';
           }
    }

?>
```

```
<?php
class MySQL_DB
{
    private $dbObj = null;
    private $result = null;

/* Для конструктора - адрес, имя пользователя, пароль, имя базы данных, порт,
/*а также кодировку для соединения.
/* По умолчанию используется utf8 */

    public function __construct($host, $user, $password, $base, $port = null, $charset = 'utf8')
    {
        $this->dbObj = new mysqli($host, $user, $password, $base, $port);
        $this->dbObj->set_charset($charset);
    }

/*основная и единственная функция, которая выполняет запрос и возвращает результат его работы*/

    public function query($query)
    {
```

```
        if(!$this->dbObj)
            return false;

/*очищаем предыдущий результат*/

        if(is_object($this->result))
            $this->result->free();

/*выполняем запрос*/

        $this->result = $this->dbObj->query($query);

/*если есть ошибки - выводим их*/

        if($this->dbObj->errno)
#!!!#           die("mysqli error #".$this->dbObj->errno.": ".$this->dbObj->error);
            return $this->dbObj->error;

/*если в результате выполнения запроса (например SELECT...) получены данные - возвращаем их.
/* данные всегда возвращаются в массиве, даже если запрос возвращает одну запись.*/
```

```
            if(is_object($this->result))
            {
                while($row = $this->result->fetch_assoc())
                    $data[] = $row;

                return $data;
            }

/*если результат отрицательный - возвращаем false */

            else if($this->result == FALSE)
                return false;

/*если запрос (например UPDATE или INSERT) затронул какие-либо строки - возвращаем их количество*/

            else return $this->dbObj->affected_rows;
        }
}
require_once 'DB_data.php';
```

```
$dbObj = new MySQL_DB(DB_HOST, DB_USER, DB_PWD, DB_NAME);
?>
```

```
<?php
$dbObj ->query("set character_set_client  ='cp1251'");
$dbObj ->query("set character_set_results ='cp1251'");
$dbObj ->query("set collation_connection  ='cp1251_bin'");
?>
```

```
<?php
#########################################################################
################
# Модуль выводит на экран результат поиска вхождения части слова во всей базе     #
# Полностью все данные не выводятся на экран, а только часть строк                 #
#########################################################################
################
#
require_once            'DB_Tables_Fields.php';  //!!!//
$String              =  trim($_POST[String]);
$Arr_acc              =  array();
$Row                  =  array();
$dbtable              =  $FormTypes;
//$head_color   = 'silver';
# Класс для построения таблиц по заголовкам и содержанию таблиц
require_once 'Any_Table_HTML_New.php';

# Класс для работы с базой данных MySQL, описание в теле модуля
# require_once 'DB_Class.php';
# Объект класса работы с базой данных создается внутри модуля DB_Class.php
```

```
# $dbObj = new MySQL_DB(DB_HOST, DB_USER, DB_PWD, DB_NAME);

###

# Так мы делаем софт более независимым от конкретных таблиц БД

#

# Заголовки всех таблиц веб проекта, которые появляются на экране

include_once 'Table_Headers.php'; //!!!// если уже созданы русские заголовки

#

# Заголовки всех таблиц БД проекта, на английском

# require_once 'Table_Headers_DB.php'; //!!! В этой версии загружаются из БД

# // на лету рограммой DB_Tables_Fields.php

require_once 'Header.php'; // заголовок для страниц проекта

require_once 'FormMsgs.php'; // все сообщения программ проекта

#

#

$f=0;  // номер элемента $FormType

foreach ($dbtable as $value) {

#

$headers = $Table_Headers [$FormTypes[$f]];// наименования столбцов таблице
```

```
$headers_db = $Table_Headers_DB[$FormTypes[$f]];// наименования доменов в таблице
#                                    // БД и <input name's в формах
###
# Создаем объекты класса для вывода таблиц, вызываем конструктор
# $handler     = $FormType.'_Handler'; // обработчик настоящей формы

$Table_view =  new HTML_Table($headers , $heard_color);
#
#запрос на выдачу всей таблицы

$QueryResult = $dbObj->query("SELECT * FROM $dbtable[$f]");
# возвращает массив ассоциативных массивов (см. класс DB_Class.php)
# по сути двумерный массив, поэтому используем конструкцию foreach
# это можно увидеть убрав знак # перед print см. ниже. Таким образом
# формируются массивы для вывода на экран для каждой таблицы
#
if ($QueryResult==false)
   {
    $msg = 'Нет записей в базе данных!';
   }
```

```
else
  {
   $NA  = count($QueryResult);
   //print '$NA='.$NA;
   //print '<pre>'; print_r($QueryResult); print '</pre>';
  }
$Strings_Number=0;
foreach ($QueryResult as $row    => $Arr_acc)
     {
      $i=0;
      $StringBig='';
      foreach ($headers_db as $element)
            {
           $Row          =          array_merge($Row,          array($headers[$i]          =>
$Arr_acc[$headers_db[$i]]) );
            $StringBig.=$Arr_acc[$headers_db[$i]];
          $i++;
           }
//      print "<br>$j. StringBig $j= $StringBig";
     $pos    =   strpos($StringBig, $String);
//           if ($pos === false) {echo "<br>Строка '$String' не найдена в строке
'$StringBig'"; continue; }
```

```
        if ($pos)
          {
          $Table_view  ->  AddRowAssArr($Row);
          $Strings_Number++;
         }
       }
if ($f == 0) print "<p></p><center><font color='Red' ><strong>НАЙДЕНО НА САЙТЕ СИСТЕМЫ  </strong></font></center>";
if ($Strings_Number >0)
  {
  print "<p></p><font color='Blue'><strong>ТАБЛИЦА $All_Tables_Names[$f]: КОЛИЧЕСТВО СТРОК - </strong></font>
              <font color='Red' size='4'><strong>$Strings_Number</strong></font>";
  print "<p></p><font color= \"blue\" ><strong>$All_Tables_Names[$f]</strong></font>"
       ."$Comment_to_Table_up[$f]";
        $Table_view     ->  PrintArr();
  print "<a href=\"Tables_Handler.php?FormType=$FormTypes[$f]\">
    Все данные Таблицы \"$All_Tables_Names[$f]\"</a>"
   .'<p></p>';
  }
//print "<a href=\"$FormType[$f]_Handler.php?FormType=$FormType[$f]\">
```

```
//      Все данные Таблицы \"$All_Tables_Names[$f]\"</a>"
#
$f++;                       // новое значение $FormType, новая таблица
}
#
include 'Footer.php';           // стандартная нижняя строка каждой страницы
#
?>
```

```
<?php
#######################################################
# Пример использования SHOW COLUMNS и SHOW TABLES  #
#######################################################
#
require_once        'DB_Class.php';
$Arr_acc_tables    = array();
$Arr_acc_fields    = array();
$Tables            = array();
$Fields            = array();
$Fields_1          = array();
$Work              = array();
$Table_Headers_DB  = array();
$SQL_String1       = array();
$SQL_String2       = array();
#
$Tables_DB_Name     = 'Tables_in_'.DB_NAME;
#
$QueryResult = $dbObj->query("SHOW TABLES");


$i=0;
```

```
foreach ($QueryResult as $row_tables    => $Arr_acc_tables)
    {
      $Tables[$i]   = $Arr_acc_tables[$Tables_DB_Name];
      $QueryResult  = $dbObj->query("SHOW COLUMNS FROM $Tables[$i]");

      $j=0;
      $string ='';
      $string1='';
      $string2='';
      foreach ($QueryResult as $row_fields    => $Arr_acc_fields)
          {
            $Fields[$i][$j]   = $Arr_acc_fields[Field];
            $Fields_1  [$j]   = $Arr_acc_fields[Field];
            $string.= ','  .$Fields[$i][$j];
            $string1.=","  .$Fields[$i][$j];
            $string2.=",$".$Fields[$i][$j];
            $j++;
          }
      $string1          = substr($string1,1);
      $string2          = substr($string2,1);
```

```
        $Table_Headers_DB    =  array_merge($Table_Headers_DB,array($Tables[$i] => $Fields_1));

        $SQL_String1         =  array_merge($SQL_String1       ,array($Tables[$i] => $string1 ));

        $SQL_String2         =  array_merge($SQL_String2       ,array($Tables[$i] => $string2 ));

        $Fields_1        = $Work;

        $i++;

}

$FormTypes=$Tables;

### Эту часть только при первом запуске, пока не созданы заголовки Table_Headers.php

include_once 'Table_Headers.php';

if (!$Table_Headers)

  {

   $Table_Headers   =$Table_Headers_DB;

    print '<br>***** $Table_Headers *****';

    print '<pre>';

    print_r($Table_Headers);

    print '</pre>';

  }

?>
```

```
<?php
include      'Header.php';
print "
<p></p>
<table bgcolor='#999999' border='0' height='8' width='1240' cellpadding='1' cellspacing='1' align='center'>
  <tr>
  <td align='center'  bgcolor='#FFFFFF'>
";

if (!$msq)
    print "<br><strong>Добро пожаловать ".$username.'!</strong>';
else print "<br><strong>$msq</strong>";

print "<br><img src = '/Pictures/lock_gre.gif'> ";
include 'Form_Buttons_Staff.php';
print   '<strong><font color="Silver"> ( Примеры: мен - все менеджеры
                                )
        </font></strong>';
print   "<br><img src = '/Pictures/lock_gre.gif'> ";
include 'Form_Buttons_Equip.php';
```

```
print '<strong><font color="Silver"> ( Примеры: 1. Админ, 2016 - Администрация за 2016 год
                <br>Если только дата, то за определенный период 2016-01 - январь
                )
    </font></strong>';
print "<br><img src = '/Pictures/lock_gre.gif'> ";
include 'Form_Buttons_AllDB.php';
print '<strong><font color="Silver"> ( Примеры: 1. Иман - поиск по части слова, фамилия
                2. МУП - входит в название компании клиента
                )
    </font></strong>';

print "
</td>
</tr>
</table>";
include 'Footer.php';
?>
```

```
<?php
require_once 'Header.php';
?>
<p></p>
<p></p>
<p></p>
<table bgcolor="#999999" height=20 width="99%" cellspacing="1" cellpadding="1" border="0">
<tr>
<td align="left" valign="top" bgcolor="#FFCCCC">
 <p></p>
 <p><font  color="Maroon" size="3">
<center><p><strong>ДОБРО ПОЖАЛОВАТЬ!</p></strong> </center>
<center><img src="/Pictures/lock_gre.gif">  Для входа в систему и начала работы кликнете по
одной из ссылок, расположенных ниже, в зависимости от Вашего статуса.</center>
<br>

</font>
 </p>
 <p></p>
```

```
</td>

</tr>

</table>

<p></p>

<p></p>

<p></p>

<p></p>

<p></p>

<table bgcolor="#999999" height=20 width="99%" cellspacing="1" cellpadding="1" border="0">

<tr>

<td align="left" valign="top" bgcolor="#FFFFFF">

 <p></p>

 <center><p><font  color="Grey" size="3">

<img src="/Pictures/lock_gre.gif">  <a href="/Clients/index_member.php"><font  color="Green" size="3">

 <strong>

 Вход для Пользователей</strong></font></a>

 </center>

</font>

 </p>

 <p></p>
```

```
<p></p>
 <p></p>
<center>
<p><font  color="Red" size="3">
<center><img src="/Pictures/lock_red.gif">  <a href="/Admin/index_admin.php"><font  color="Red" size="3">
<strong>
Вход для администратора системы</strong></font></a>
</center>
</font>
</p>
</center>
<p></p>
</td>
</tr>
</table>
<?php require_once 'Footer.php'; ?>
```

```
<?php
#########################################################################
################
# Модуль производит расширенный поиск по полям - наименование документа,
#
# можно унифицировать программу для любых таблиц)                       #
# При клике на ссылке "расширенный поиск" Form_Buttons.php вызывается
обработчик  #
# Tables_Handler.php с меню расширенного поиска Form_Buttons_Equip.php
затем     #
# Обработку поиска производит модуль Equip_Search                       #
#                                                                       #
#########################################################################
################
#
$Arr_w        = array();
$Row_w        = array();
$FormType     ='3_docs';    // таблица документов
$Staff_Search = true;     //!!!///
$dbtable      = $FormType;
#
# в ранних версиях был файл require_once 'Table_Headers_DB.php';
# Заголовки всех таблиц БД проекта, на английском формируются динамически
```

```
require_once 'DB_Tables_Fields.php';
#
# Класс для построения таблиц по заголовкам и содержанию таблиц
require_once 'Any_Table_HTML_New.php';

# Класс для работы с базой данных MySQL, описание в теле модуля
require_once 'DB_Class.php';
#
# Заголовки всех таблиц веб проекта, которые появляются на экране
include_once 'Table_Headers.php';
#
# Так мы делаем софт более независимым от конкретных таблиц БД
#
$headers    = $Table_Headers   [$FormType];// наименования столбцов таблице
$headers_db = $Table_Headers_DB[$FormType];// наименования доменов в таблице
#                                  // БД и <input name's в формах
# Создаем объекты класса для вывода таблиц, вызываем конструктор
#$handler     = $FormType.'_Handler';      // обработчик настоящей формы

#
```

```
# Объект класса работы с базой данных создается внутри модуля DB_Class.php
//$dbObj = new MySQL_db(DB_HOST, DB_USER, DB_PWD, DB_NAME);
#
#include 'Client_form_valid.php  // проверяет введенные данные

#****************************** Поиск по ключам ******************************
# то только записи соответствующие ввденным ключам будут найдены и выданы на экран
#
if (!($_POST["name"])) // наименование, можно сокращенно
    $KeyValue1 = '%'; // может быть пустым
else
    {
    $KeyValue1 = trim($_POST["name"]);
    $KeyValue1 = '%'.$KeyValue1.'%';
    }
#
if (!($_POST["year"])) // год - месяц - число
    $KeyValue2 = '%'; // да или нет
else // может быть пустым
```

```
      {
       $KeyValue2    = trim($_POST["year"]);
       $KeyValue2    = '%'.$KeyValue2.'%';
      }
#
if    (!($_POST["content"]))
      $KeyValue3    = '%';
else
      {
       $KeyValue3    = trim($_POST["content"]);// может быть пустым
                                  // краткое содержание
       $KeyValue3    = '%'.$KeyValue3.'%';  // 2 - например
      }
#
if (($KeyValue1=="%")&&($KeyValue2=="%")&&($KeyValue3=="%"))
  {
  $msq  ='<font color="Red"> Ошибка! Введите данные для поиска</font>';
  include_once 'Entry_Search.php';        //!!!!!//
  exit;
  }
# поиск по нескольким ключам
```

```
else
  {
    $KeyName0  =  $headers_db[0];
    $KeyName1  =  $headers_db[4];
    $KeyName2  =  $headers_db[6];
    $KeyName3  =  $headers_db[5];
  $$FormType = $dbObj->query("SELECT * FROM  $dbtable
                              WHERE $KeyName1 LIKE '$KeyValue1'
                              AND   $KeyName2 LIKE '$KeyValue2'
                               AND   $KeyName3 LIKE '$KeyValue3'
                              ORDER BY '$KeyName0' ");

  }
#
### Это структура таблицы документа БД. Имена полей справа. Слева имена столбцов на русском
#
/*

   '3_Docs' => Array
      (
```

```
        'Код документа'                ,  # [0] => Docs_Id
        'Код отправителя'              ,  # [1] => Client_Id
        'Наименование организации'     ,  # [2] => Client_Nm
        'Исходящий номер'              ,  # [3] => Docs_Out
        'Наименование'                 ,  # [4] => Docs_Nm
        'Описание/Характеристика'      ,  # [5] => Docs_Detail
        'Дата поступления'             ,  # [6] => Docs_Time
        'Ссылка на документ'           ,  # [7] => Docs_Link   Ссылка на документ
        'Статус документа'             ,  # [8] => Docs_Stat   Проставляется системой
    ),

*/
# возвращает массив ассоциативных массивов (см. класс DB_Class.php)
# по сути двумерный массив, поэтому используем конструкцию foreach
# это можно увидеть убрав знак # перед print см. ниже. Таким образом
# формируются массивы для вывода на экран, даже если таблица состоит
# из одной строки (как в случае поиска)
#
if ($$FormType==false)
  {
 $msq  ='<font color="Red"> Запись не найдена в таблице</font>';
```

```
 include_once 'Entry_Search.php';                    //!!!!!//
 exit;
  }
else
  {
   $NC  = count($$FormType);
//  print '$NC='.$NC;
//  print '<pre>'; print_r($$FormType); print '</pre>';
  }
$head_color   = 'silver';
$Table_view   =  new HTML_Table($headers , $head_color);
foreach ($$FormType as $row    => $Arr_w)
     {
      $i=0;
      foreach ($headers_db as $element)
            {
            $Row_w        =        array_merge($Row_w,        array($headers[$i]        =>
$Arr_w[$headers_db[$i]]) );
            $$headers_db[$i] = $Arr_w[$headers_db[$i]];
          $i++;
           }
```

```
    $Table_view ->  AddRowAssArr($Row_w);

     }

#

include'Header.php';              // общий заголовок для всех страниц проекта

print "<p><font color='red'><strong>НАЙДЕНЫ ДОКУМЕНТЫ</strong> $msg1 </font></p>";

$Table_view -> PrintArr();              // таблица 'Staff' - все назначения на текущий момент

include 'Form_Buttons_Equip.php';

print "<p><a href=\"Tables_Handler.php?FormType=$FormType\">Просмотр всей таблицы документов</a>";

include 'Footer.php';              // стандартная нижняя строка каждой страницы

#

?>
```

```
<!--без таблиц //-->

<p></p>

 <form action="Tables_Handler.php?FormType=<?php print $FormType?>" method=post>

 <!--form action="<?php //print $FormType;?>_Handler.php" method=post/-->

  <strong>Введите Id код для удаления, поиска или обновления </strong>

  <input type="text" name ="<?php print $headers_db[0];?>" size=10 maxlength=10

                value ="<?php print $$headers_db[0];?>">

  <INPUT TYPE="submit" NAME="Delete" VALUE="Удалить">

   

  <INPUT TYPE="submit" NAME="Search" VALUE="Найти">

   

  <INPUT TYPE="submit" NAME="Update" VALUE="Обновить">

   

 </form>
```

```
<!--без таблиц //-->
<br><strong>Поиск по сайту по вхождению части слова </strong>
<p></p>
 <form name="Db_Search" action="DB_Search.php" method=post>
  <input type="text"   name="String" size=10 maxlength=80   value="">
  <INPUT TYPE="submit" NAME="SEARCH" VALUE="   Найти  ">
   
 </form>
```

```
<!--без таблиц //-->
<br><strong>Введите данные для поиска по ключам документов</strong>
<p></p>
 <form name="del" action="Equip_Search.php?FormType=<?php print $FormType;?>" method=post>
   &nbspНаименование документа (часть слова)  
   <input type="text"  name="name" size=10 maxlength=8  value="">
    
     Год-Месяц-Дата или Год-месяц, Год   
   <input type="text"  name="year" size=10 maxlength=10  value="">
    
     Содержание (часть слова)  
   <input type="text"  name="content" size=10 maxlength=10  value="">
    
   <INPUT TYPE="submit" NAME="SEARCH" VALUE="  Найти  ">
 </form>
```

```
<!--без таблиц //-->
<h4>Введите данные для поиска по сайту&amp;nbsp;</h4>
<p></p>
  <form name="Db_Search" action="tst2.php" method=post>
    <input type="text"   name="String" size=10 maxlength=80   value="">
    <INPUT TYPE="submit" NAME="SEARCH" VALUE="   Найти  ">
     
  </form>
```

```
<!--без таблиц //-->
<br><strong>Введите данные для поиска по ключам. </strong>
<p></p>
<form name="del" action="Staff_Search.php" method=post>
   &nbspДолжность специалиста  
   <input type="text"   name="post" size=10 maxlength=8    value="">
    
     Наличие ВО    
   <input type="text"   name="sert" size=10 maxlength=10  value="">
    
     Год выдачи диплома  
   <input type="text"   name="year" size=10 maxlength=10   value="">
   <INPUT TYPE="submit" NAME="SEARCH" VALUE="   Найти    ">
    
 </form>
```

```
<?php
#########################################################################
################
# Модуль выводит на экран форму для добавления данных в любую таблицу
#
#########################################################################
################
if        ($_GET['formtype'])          //!!!//
 $FormType = $_GET['formtype'];
require_once      'DB_Tables_Fields.php';  //!!!//
# В этой версии формируется программой 'DB_Tables_Fields.php'
//require_once     'FormTypes.php';        //!!!//
#
require_once      'FormMsgs.php';
#
# Форма для добавления, удаления, обновления данных по подразделениям,
# сотрудникам, приборам, изделиям, результатам испытаний и пр.
# Формируются исходные данные для формы и вызывается универсальный
# построитель форм такого вида Universal_Forms_Builder.php
###
# В случае вызова модуля для заполнения включаются эти include-ы
include_once   'Header.php';              // Общий заголовок веб-страниц проекта
```

```
include_once   'Table_Headers.php';          // Заголовки таблиц и форм таблиц

# В этой версии Table_Headers_DB формируется программой 'DB_Tables_Fields.php'

//include_once   'Table_Headers_DB.php';     // Заголовки таблиц базы данных

include_once    'Any_Table_HTML_New.php';      // Класс для построения таблиц по

                                    // заголовкам и содержанию

#

$handler      = 'Tables_Handler.php?FormType='.$FormType;

                                    // обработчик настоящей формы

//$handler      = $FormType.'_Handler';      // обработчик настоящей формы

$headers      = $Table_Headers   [$FormType];// наименования столбцов таблиц

$headers_db      =  $Table_Headers_DB[$FormType];// наименования доменов в таблице

$head_color   = "silver";                  // Staff БД и input name's в формах

$form_headers = array(' Наименования полей',

                ' Заносимые или обновляемые данные'

                );

# Создаем объекты класса для вывода таблиц, вызываем конструктор

#

$Form_view   =  new HTML_Table($form_headers  , $head_color);

# Заголовок формы страницы
```

```
print    '<p><font color= "red" ><strong>'
           .$FormMsgs[$FormType][0].
           '</strong></font>
         </p>';                          // Заголовок формы страницы
#
# построитель простых форм такого вида Universal_Forms_Builder.php
#
include 'Universal_Forms_Builder.php';
?>
<p></p>
</body>
</html>
```

```
<html>
 <head>
  <title>ИС РЕГИСТРАЦИИ И ПОИСКА ДОКУМЕНТОВ УФИНП АДМИНИСТРАЦИИ ГРЕМЯЧИНСКОГО МР"
  </title>
  <style>
  a {
  text-decoration: none; /* Отменяем подчеркивание у ссылки      */
  }
  A IMG
  {
  border: none;          /* Убираем рамку в изображениях-ссылках */
  }
  body
  {
  background: #E0FFFF;  /* Цвет фона                    */
  color: Maroon;        /* Цвет текста                  */
  }
  </style>
 </head>
 <body>
```

```
<table border="0" height=5 width="99%" cellpadding="1" cellspacing="1">
 <tr align="center">
  <td colspan="9">
   <img src="/Pictures/Logotip1.jpg"><img src="/Pictures/Logotip2.jpg">
  </td>
 </tr>
 <tr><td width="20%" align="center" bgcolor="blue">
  <a href="/index.php"><font color="Gold" size="3"><strong>О ПРОЕКТЕ</strong></font></a>
 </td>
 <td width="20%" align="center" bgcolor="blue">
  <a href="/Entry_System_Member.php"><font color="Gold" size="3"> <strong>ВХОД </strong></font></a>
 </td>
 <td  width="20%" align="center" bgcolor="blue">
  <a href="/Entry_Search.php"><font color="Gold" size="3"> <strong>ПОИСК</strong></font></a>
 </td> )

 <td  width="20%" align="center" bgcolor="blue">
  <a href="/About.php"><font color="Gold" size="3"> <strong>О НАС</strong></font></a>
```

```
    </td>
   </tr>
  </table>
  <table bgcolor="#999999" height=10 width="99%" cellspacing="1" cellpadding="1" border="0">
   <tr>
    <td align="left" valign="top" bgcolor="Olive">
     <center><font color="Gold" size="3">УПРАВЛЕНИЕ ФИНАНСОВ И НАЛОГОВОЙ ПОЛИТИКИ АДМИНИСТРАЦИИ ГРЕМЯЧИНСКОГО МУНИЦИПАЛЬНОГО РАЙОНА</font></center>
    </td>
   </tr>
  </table>
</body>
</html>
```