МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«РОСТОВСКИЙ ГОСУДАРСТВЕННЫЙ ЭКОНОМИЧЕСКИЙ УНИВЕРСИТЕТ» (РИНХ)

ФАКУЛЬТЕТ КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

Кафедра Информационных систем и прикладной информатики

Допустить к защите:

И.о. зав. кафедрой д.э.н.
_____ С. М. Щербаков

«17» февраля 2020 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

на тему:

«Разработка системы голосового управления компьютером для людей с ограниченными возможностями в Ростовском филиале ООО «Юдевелопмент»

Выполнил
студент группы 341 – ПИZS А.И. Исаев

Направление 09.03.03 «Прикладная информатика»
Профиль 09.03.03.01 «Прикладная информатика в экономике»

Руководитель выпускной квалификационной работы
к.э.н., доцент И.И. Мирошниченко

Ростов–на–Дону, 2020 г.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«РОСТОВСКИЙ ГОСУДАРСТВЕННЫЙ ЭКОНОМИЧЕСКИЙ УНИВЕРСИТЕТ» (РИНХ)

ФАКУЛЬТЕТ КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

Кафедра Информационных систем и прикладной информатики

Утверждаю:

И.о. зав. кафедрой д.э.н.

_____ С. М. Щербаков

«03» февраля 2020 г.

ЗАДАНИЕ НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ

Студенту Исаеву Артему Игоревич Код 02

Тема: «Разработка системы голосового управления компьютером для людей с ограниченными возможностями в Ростовском филиале ООО «Юдевелопмент»

Утверждено приказом РГЭУ № 783/4к от 03.10.2019 г.

Срок сдачи студентом законченного проекта 10.02.2020 г.

Исходные данные по проекту: материалы преддипломной практики; нормативно-методическая документация; методические указания, разработанные кафедрой, предпроектное обследование предприятия.

Содержание выпускной квалификационной работы (по разделам):

а) Аналитическая часть.

б) Проектная часть.

в) Обоснование экономической эффективности проекта.

Перечень графического материала:

- организационная структура предприятия;
- контекстная диаграмма деятельности компании;
- диаграмма декомпозиции контекстной диаграммы деятельности;
- диаграмма информационной модели программного продукта;
- диаграмма вариантов использования;
- диаграмма активности.

Дата выдачи задания 03.10.2019 г.

Календарный график выполнения ВКР

Наименование этапов ВКР	Срок выполнения этапов проекта	Примечание
1. Аналитическая часть задачи «Разработка системы голосового управления компьютером для людей с ограниченными возможностями в Ростовском филиале ООО «Юдевелопмент»	26.12.2019 г.	
2. Проектная часть задачи «Разработка системы голосового управления компьютером для людей с ограниченными возможностями в Ростовском филиале ООО «Юдевелопмент»	10.01.2020 г.	
3. Обоснование экономической эффективности задачи «Разработка системы голосового управления компьютером для людей с ограниченными возможностями в Ростовском филиале ООО «Юдевелопмент»	31.01.2020 г.	
4. Оформление выпускной квалификационной работы и сдача ее на кафедру	10.02.2020 г.	

Руководитель проекта _____ И.И. Мирошниченко

Задание принял к исполнению _____ А.И. Исаев

Реферат

Выпускная квалификационная работа: 95 страниц, 23 рисунков, 3 таблицы, 3 приложения, 26 источников

БИЗНЕС ПРОЦЕССЫ, ПРОГРАММНЫЙ ПРОДУКТ, АВТОМАТИЗАЦИЯ, ЭКОНОМИЧЕСКАЯ ЭФФЕКТИВНОСТЬ, РАСПОЗНАВАНИЕ РЕЧИ

Целью выпускной квалификационной работы является разработка программного продукта для голосового управления компьютером на операционной системе Windows 8 и выше.

Используемый метод решения – индивидуальное проектирование.

В качестве проектных инструментов и средств для создания программного продукта используются Microsoft Visual Studio, Google console, Google SpeachToText, Microsoft Access.

Результатом решения поставленной в проекте задачи, являться программный продукт голосового управления компьютером, предоставляющий возможность взаимодействия с компьютером на базе операционной системы Windows 8 и выше при помощи голосовых команд. Результатная информация представлена в виде экранных форм, сообщений.

Разработанный программный продукт предназначен для использования людьми с ограниченными возможностями.

СОДЕРЖАНИЕ

Введение

Тема выпускной квалификационной работы: «Разработка программного продукта для голосового управления компьютером для людей с ограниченными возможностями в благотворительных целях, на базе предприятия ООО «ЮДЕВЕЛОПМЕНТ».

В настоящее время технологического развития человечество сделало большой шаг в развитии информационных технологий. С уровнем роста технологий возрос и уровень автоматизации обыденных вещей. Автоматизации подвергаются практически все сферы деятельности человека. Хороший пример автоматизации - это двери в супермаркете. Автоматизация процесса открытия дверей позволила улучшить процесс покупок, а именно люди с тяжелыми пакетами не тратят время и силы на то, чтобы открыть дверь. Некоторые технологии придуманы для людей с ограниченными способностями. Одним из быстро развивающихся направлений является разработка речевого интерфейса, который позволяет управлять различными устройствами. В основе голосового управления лежит технология распознавания речи. В ней задействованы достижения различных областей: от компьютерной лингвистики до цифровой обработки сигналов.

Точность результатов напрямую зависит от того, насколько хорошо система определяет произнесённые звуки. Для этого достаточно точным и полным должен быть фонетический алфавит, с которым она работает. В итоге данная технология способна распознавать до 90–95% речи [11].

Для людей с ограниченными возможности существует большое количество программного обеспечения для пользования компьютером, но среди этих программных существует малое количество голосовых помощников. Программы голосового управления решают задачу взаимодействия пользователя с компьютером и могут использоваться для облегчения работы с устройством [11].

Программа предназначена для голосового управления компьютером, программного изменения состояния компьютера.

Программа является благотворительной и не является автоматизацией бизнес-процессов предприятия. Но можно утверждать, что благотворительность - это неотъемлемая часть предприятия. Благотворительность занимает важное значение в деятельности предприятия. Она способствует рекламе услуг предприятия, что в дальнейшем может привести к сотрудничеству с крупными клиентами.

Выпускная квалификационная работа содержит введение, три главы и заключение. В первой главе проводится анализ выбранной предметной области. Также, в этой главе приводится постановка задачи, и описываются требования к программному продукту. В завершении главы проводится обоснование выбора технического, информационного и программного обеспечения задачи.

Вторая глава ВКР является практической и включает в себя описание программного кода для разработанного программного продукта. Завершает вторую главу описание примера работы сайта.

В третьей главе приведены расчеты для обоснования экономической эффективности.

Теоретической основой для написания работы послужили публикации в российских и зарубежных изданиях.

В выпускной квалификационной работе использованы следующие методы: наблюдение, анализ, моделирование, разработка, внедрение.

Результатом работы над выпускной квалификационной работой является программный продукт голосового управления компьютером.

1 АНАЛИТИЧЕСКАЯ ЧАСТЬ

1.1 Технико–экономическая характеристика предметной области

1.1.1 Характеристика предприятия

Полное наименование исследуемой организации – ООО «ЮДЕВЕЛОПМЕНТ» – Фрагменты Устава представлены в приложении А, на рисунках А.1 – А.4.

Юридический адрес: 353680, Краснодарский край, Ейский р–н, г.Ейск, ул. Грушовая, д. 10, Фактический адрес: 344082, РФ, Ростовская обл., г. Ростов–на–Дону, ул. Береговая, д. 8 (БЦ “Риверсайд“), оф. № 106

Основная деятельность предприятия заключается в оказании IT услуг.

Виды деятельности предприятия:

- разработка компьютерного программного обеспечения;
- консультативная деятельность в области программного обеспечения;
- деятельность, связанная с использованием вычислительной техники и информационных технологий;
- деятельность по созданию и использованию баз данных и информационных ресурсов;
- ремонт компьютеров и периферийного компьютерного оборудования.

Сведения о видах деятельности предприятия представлены в приложении А, на рисунке А.2.

1.1.2 Описание подразделений предприятия и видов их деятельности

Организационная структура ООО «ЮДЕВЕЛОПМЕНТ» представлена на рисунке 1.1.

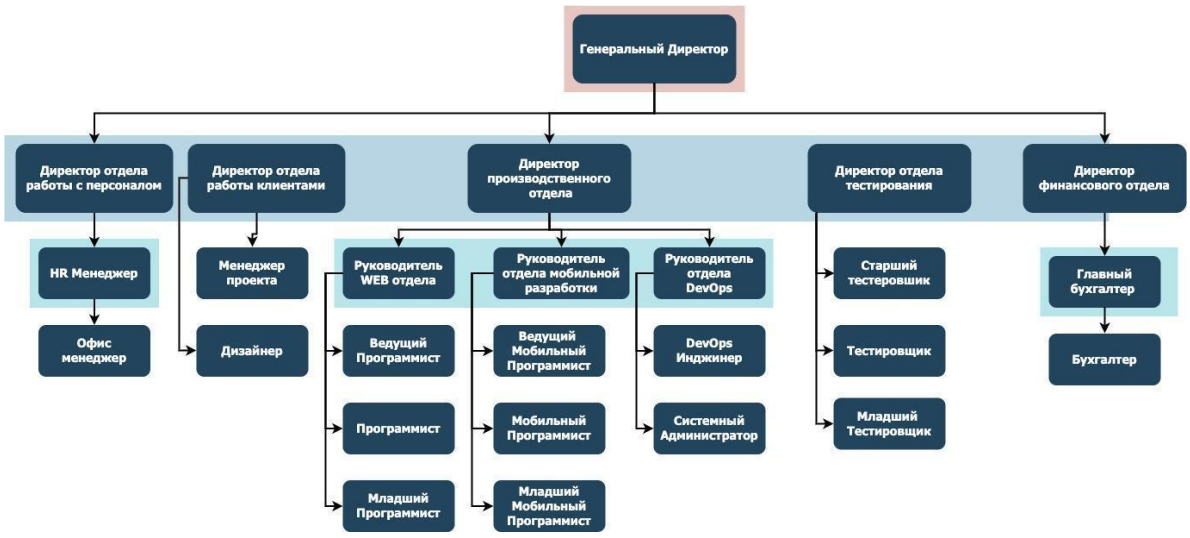


Рисунок. 1.1 – Организационная структура ООО «ЮДЕВЕЛОПМЕНТ»

Штатное расписание ООО «ЮДЕВЕЛОПМЕНТ» представлено в приложении Б.

В обязанности генерального директора входит принятие решений развития компании и привлечения клиентов. Главному директору подчиняются 5 отделов компании: отдел работы с персоналом, отдел работы с клиентами, отдел производства, отдел тестирования, финансовый отдел.

У каждого отдела есть свой руководитель который имеет в подчинении сотрудников. В обязанности руководителя отдела входит контроль подчиненных и направление развития своего отдела.

Разберем отдел по работе с персоналом. Руководитель отдела имеет в подчинении HR менеджеров и офис менеджеров. В обязанности HR менеджера входит обеспечение комфортных условий работы сотрудником компании. Они

ведут учет техники, развивают офис, занимаются организацией командных походов, решают вопросы, которые возникают у сотрудников, оформляют отпуска и больничные. В обязанности офис менеджера входит обеспечение офиса. Эти сотрудники организовывают доставку воды в офис, вызывают рабочих для починки чего–либо, рассаживают новых сотрудников на выделенные места, следят за чистотой в офисе и работоспособностью техники.

Отдел работы с клиентам отвечает за коммуникацию компании и клиентов. Проектные менеджеры являются руководителем проектной команды и в их обязанности входит:

- получение требования клиента;
- общение с клиентом;
- решение вопросов которые возникают у клиента;
- организация проектной деятельности;
- формирование задач на неделю для разработчиков;
- формирование отчетов для клиентов;
- формирование отчетов для бухгалтерии.

Дизайнеры находятся в отделе работы с клиентами потому, что часто помогают проектным менеджерам вести проект. Дизайнер обязан понимать предметную область проекта, понимать требования заказчиков, понимать развитие проекта через 2 года, разработка макетов, прототипов, логотипов, скетчей, диаграмм, презентаций.

Производственный отдел является самым большим в компании и делится на 3 направления: разработка веб-приложений, разработка мобильных приложений,development and operations (DevOps).

В обязанности руководителей отдела входит распределение нагрузки среди разработчиков и принятие решение какие технологии нужно использовать. Они дают предварительную оценку новых проектов.

В обязанности программиста входит:

- написание программного кода;

- написание документации кода;
- решение проблем заказчика в проекте;
- выстраивание архитектуры приложения;
- выстраивание архитектуры баз данных;
- проведение демо заказчику если это требуется;
- написание отчетов о проделанной работы менеджеру проекта.

Отдел тестирование занимается тестированием программного продукта. Руководитель отдела распределяет нагрузку среди подчиненных, решает вопросы связанные с отделом. Руководителю отдела подчиняются тестировщики.

В обязанности тестировщика входит:

- тестирование задач;
- поиск уязвимостей проекта;
- проведение регрессионного тестирования;
- проведение нагрузочного тестирования;
- помощь менеджеру проекта в формировании задач и отчетов;
- написание юнит-тестов;
- ведение документации проекта.

Финансовый отдел отвечает за бухгалтерию. Руководитель отдела отчитывается каждый месяц перед генеральным директором о проделанной работе. У него в подчинении есть бухгалтера. В их обязанности входит: выдача заработной платы, оформление больничных листов, оформление отпусков, расчет премии и т.д.

Самый крупный отдел компании - это производственный отдел. Разберем несколько бизнес процессов этого отдела.

Для наглядного представления о видах предпринимательской деятельности предприятия, используем методологию функционального моделирования IDEF0. В методологии IDEF0 рассматриваются логические отношения между работами и потоком работ, а не временная

последовательность. Благодаря этому можно понять взаимосвязь бизнес процессов, общего и подробного уровня исполнения. На рисунке 1.1 представлена контекстная диаграмма деятельности компании.

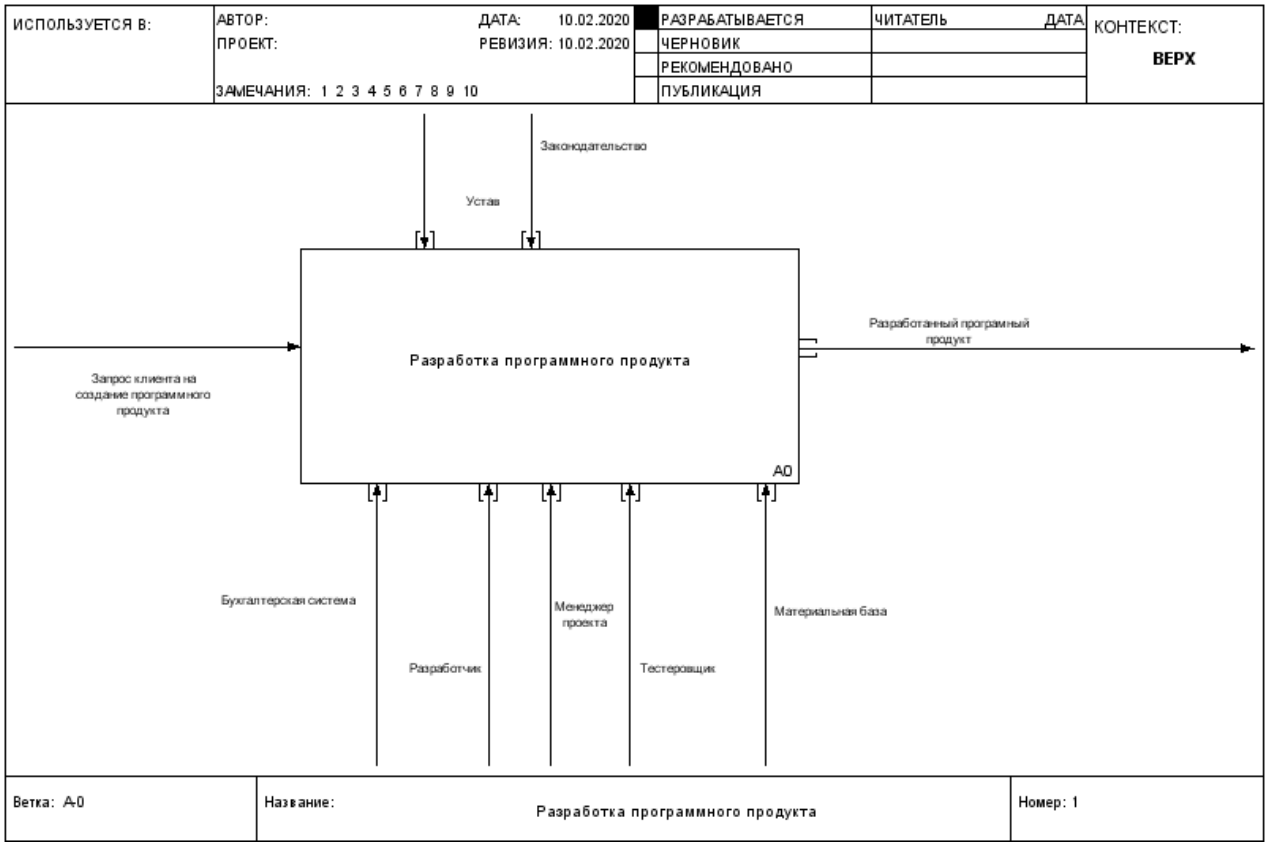


Рисунок 1.2 - Контекстная диаграмма деятельности компании

На диаграмме видно, что бизнес-процесс разработки программного продукта на входе имеет запрос на создание программного продукта. Так же использует механизмы: бухгалтерская система, разработчик, менеджер проекта, тестировщик и материальная база. Также можно увидеть, что бизнес-процесс руководствуется уставом и законодательством.

На диаграмме мы видим основные потоки бизнес процессов, которые не отражают многообразия деятельности компании. В связи с этим, нам необходимо декомпозировать диаграмму деятельности. На рисунке 1.3 представлена диаграмма декомпозиции контекстной диаграммы деятельности.

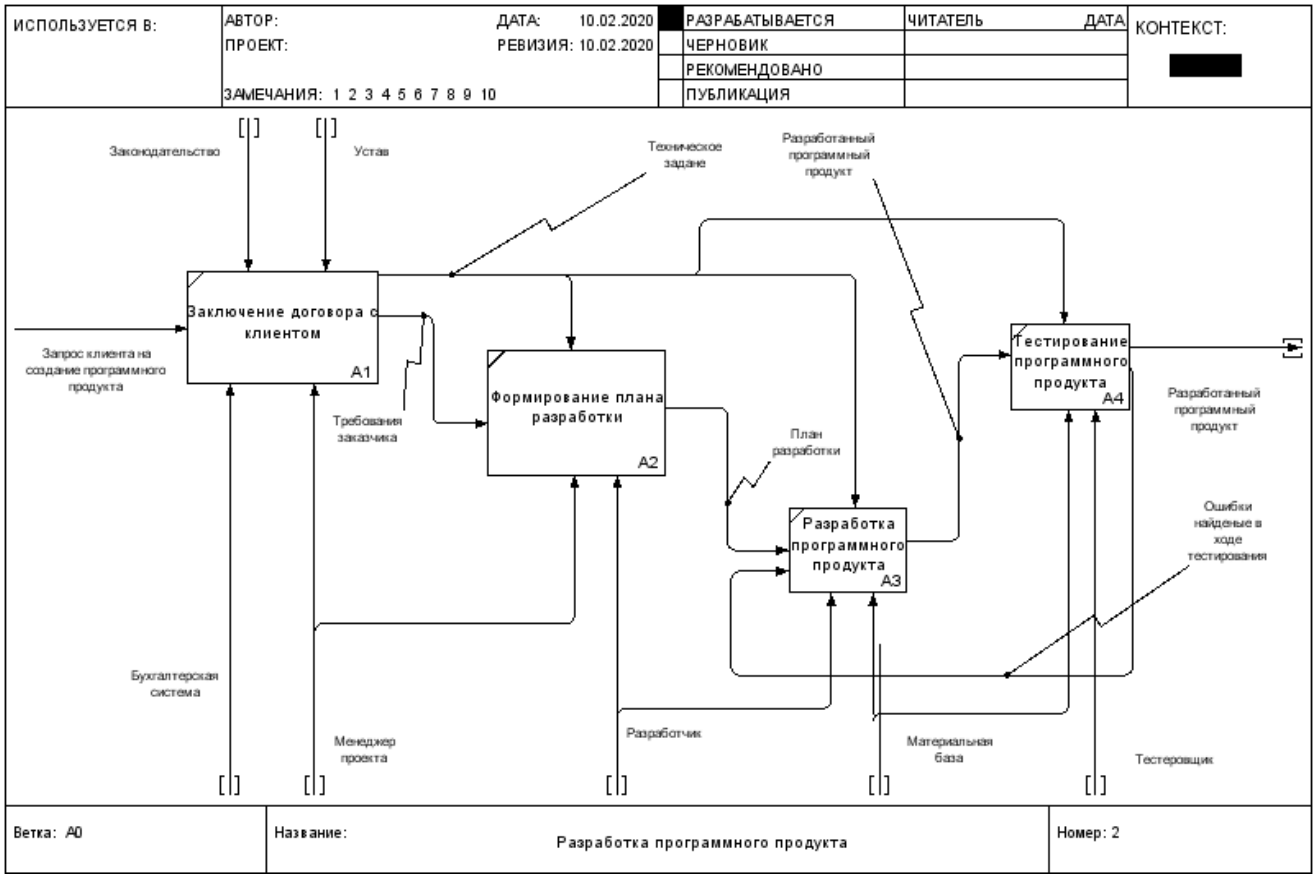


Рисунок 1.3 – Диаграмма декомпозиции контекстной диаграммы деятельности

На диаграмме мы можем видеть детализированный процесс разработки программного продукта. На входе мы имеем запрос клиента на создание программного продукта. Первый этап — это заключение договора. Этим процессом занимается менеджер вместе с бухгалтерской системой, руководствуясь законодательством и уставом предприятия. После заключения договора формируется техническое задание и формируется план разработки. Руководствуясь техническим заданием менеджер и разработчик формируют план разработки. После этого начинается процесс разработки. В нем участвует разработчик, разработка ведется на компьютерах (материальной базе) предприятия. После разработки разработанный программный продукт отдается на тестирование. В процессе тестирования участвует тестировщик. Если в ходе тестирования были найдены ошибки, то разработчик исправляет ее и отдает на

тестирование повторно. В случае если ошибок не найдено, программный продукт считается разработанным.

Кроме коммерческой разработки на предприятии существует благотворительная разработка. Контекстная диаграмма разработки программного продукта в благотворительных целях представлена на рисунке 1.4.

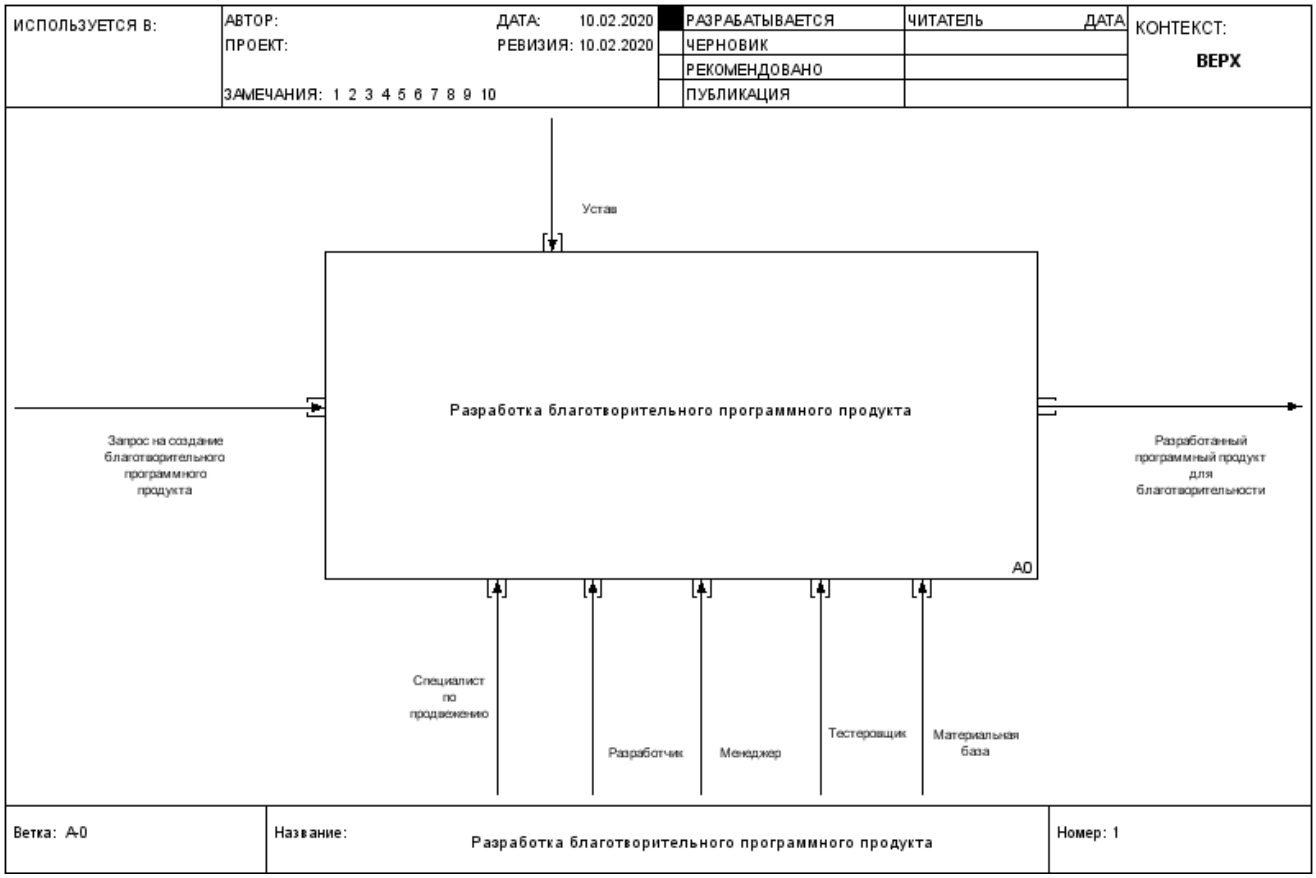


Рисунок 1.4 - Контекстная диаграмма разработки программного продукта в благотворительных целях

Как мы видим, немного отличается от контекстной диаграммы разработки программного продукта, а именно наличием специалиста по продвижению. Декомпозируем бизнес-процесс разработки программного продукта в благотворительных целях. На рисунке 1.5 представлена диаграмма декомпозиции контекстной диаграммы разработки программного продукта в благотворительных целях.

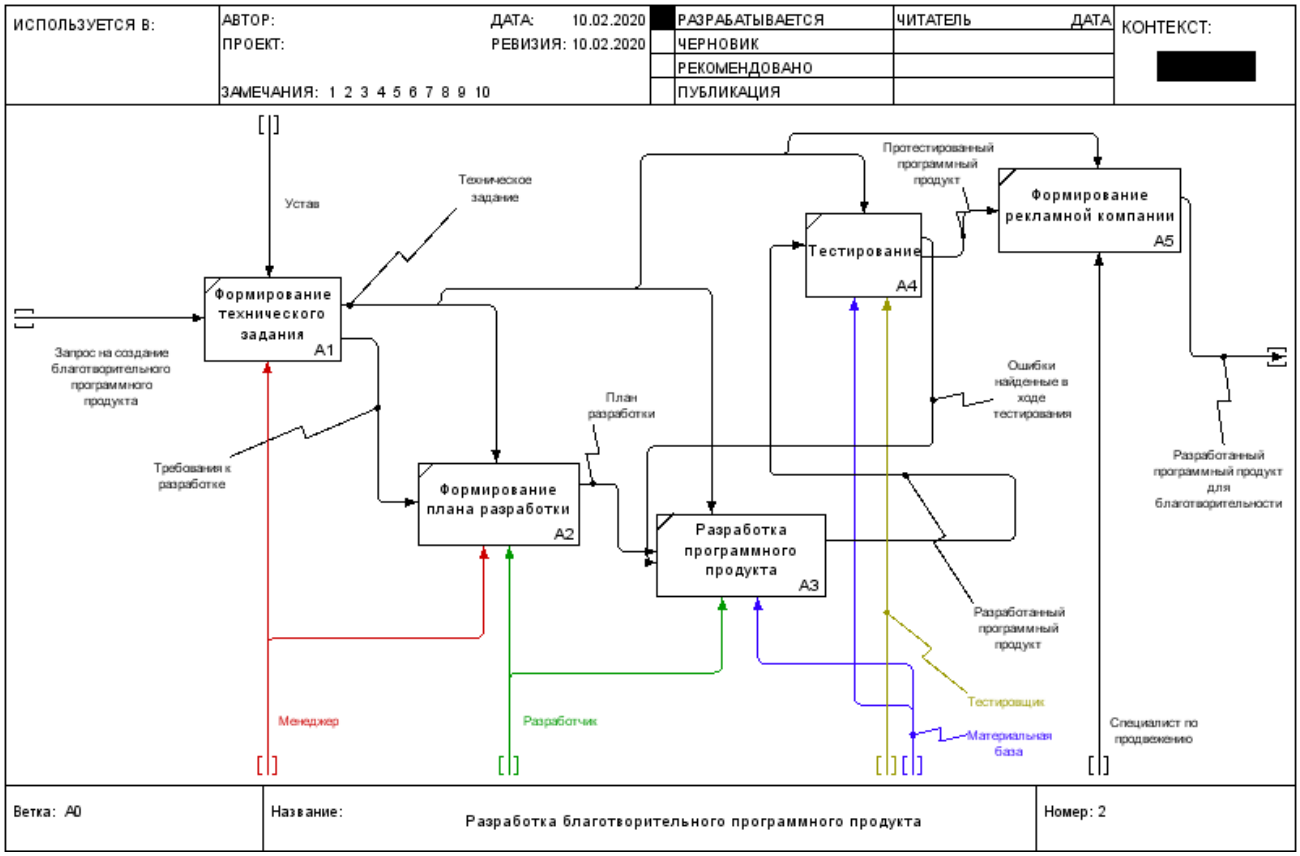


Рисунок 1.5 - Диаграмма декомпозиции контекстной диаграммы разработки программного продукта в благотворительных целях

Основные бизнес процессы почти такие же, как и при разработке коммерческого программного продукта. Отличие в том, что при разработке в благотворительных целях не заключается договор с клиентом. Также присутствует специалист по продвижению. После процесса разработки он формирует рекламную кампанию и после этого считается что программный продукт завершен.

1.2 Экономическая сущность задачи

Поскольку разработанный программный продукт является благотворительным программным обеспечением, то для оценки экономической сущности задачи автоматизации следует провести сравнение существующих решений данной задачи с предлагаемым решением.

Для сравнения рассмотрим схожий программный продукт Cortana [16] от Microsoft. Для запуска этого ассистента понадобится лицензионная версия Windows 10 PRO, стоимостью от 10 000 рублей. Cortana поддерживает только английский язык, в то время как разработанная программа поддерживает русский. Для доступа всех возможностям Cortana требуется учетная запись Microsoft с синхронизированными настройками. На данный момент Cortana не поддерживается компанией Microsoft. Cortana не имеет возможности создания своих голосовых команд.

Разработанный программный продукт может запускаться на версии Windows 8 и выше, на которой установлен .NET Framework версии 4.0, имеющий доступ в интернет. Распространяется бесплатно, развивается и имеет постоянную поддержку. Умеет работать с интерфейсом ОС и обладает возможностью создавать собственные голосовые команды.

Для пользователей разработанного программного обеспечения экономическая выгода очевидна. Бесплатное ПО выделяется своими возможностями.

Для предприятия так же существует экономическая ценность разработки данной программы. Распространяя программный продукт на бесплатной основе в благотворительных целях, предприятие получает рекламу и расширяет свое влияние на рынок десктопных приложений. Благодаря рекламе можно сформировать и укрепить положительное представление потребителя о качестве и ценных свойствах товаров и услуг.

Для крупных IT компаний большое значение имеет их имидж на рынке. Положительные отзывы о копании способствуют привлечению новый клиентов, в том числе и крупных игроков на рынке. Для этих целей компании стараются заявить о себе на мировом уровне. Компания сможет заявить о себе на рынке нативных приложений для ОС Windows. Это расширяет спектр оказываемых услуг компании, что привлечет новых клиентов этой отрасли.

Долгосрочное сотрудничество с крупными клиентами приносит большие инвестиции и помогает компании разрастаться, занимая новые ниши на IT рынке.

Разработанный программный продукт является одним из шагов в развитии исследовательского отдела компании. Изучение и проектирование нейронных сетей и искусственного интеллекта предоставляют возможности для создания инструментов, увеличивающих эффективность бизнеса.

Программный продукт разработан для людей с ограниченными возможностями. Его целью является помощь людям в управлении компьютером. Поскольку программный продукт является благотворительным, это означает, что его можно бесплатно установить на персональный компьютер.

Для пользователей программного продукта экономическая эффективность заключается в отсутствии платной лицензии. Это означает что пользователи могут установить программный продукт бесплатно.

Для компании экономическая эффективность заключается в приобретении репутации компании, которая занимается благотворительностью, что в свою очередь привлекает внимание к деятельности компании.

1.3 Обоснование необходимости и цели использования вычислительной техники для решения задачи

Программный продукт являться автоматизированной системой, состоящей из программных модулей и электронных документов, объединенных в единую систему. Создание программного продукта голосового управления компьютером невозможно без применения современных программных средств.

В разработке программного продукта использовано следующее программное обеспечение:

- операционная система Windows 10 Professional – операционная система семейства Windows, разработанная компанией Microsoft;

- Microsoft Office – комплекс программного обеспечения, способный работать с различными типами электронных документов;
- Microsoft Visual Studio – среда разработки, позволяющая создать программный продукт, используя язык C#. Помимо редактора кода в программу встроен мощный инструмент для отладки приложения;
- Total Commander – файловый менеджер, использующий графический интерфейс, для предоставления, редактирования, создания информации о файлах, папках и путях нахождения необходимых данным, хранящимся на персональном устройстве;
- Google Console – инструмент Google для управления сервисами Гугл, в частности SpeachToText;
- .NET Framework – фреймворк расширяющий возможности языка C#.

Так как программный продукт работает на операционной системе Windows, то для его разработки необходима такая же операционная система.

Для удобства работы с документами необходим пакет программ Microsoft Office. Для написания программного кода необходима среда разработки Microsoft Visual Studio. Она содержит инструменты для работы с программными продуктами на платформе Windows и позволяет писать программный код на языке C#. Microsoft Visual Studio содержит мощный инструмент отладки программного продукта, что выделяет эту среду разработки среди прочих.

Программный продукт работает на технологии распознавания голоса SpeachToText от компании Google. Для настройки этой технологии необходима консоль управления Google Console.

Для разработки под операционную систему Windows используется .NET framework, для языка программирования C#. Он содержит множество инструментов, упрощения написания кода.

Это программное обеспечение является наиболее подходящим для разработки программного продукта голосового управления компьютером под операционной системой Windows.

1.4 Постановка задачи

1.4.1 Цели и назначение автоматизированного варианта решения задачи

Целью автоматизированного решения задачи является предоставление инструмента, для взаимодействия с компьютером, людям с ограниченными возможностями.

Автоматизация позволит сократить время поиска нужной информации, а также, поможет в освоении компьютера людям, обладающими слабыми техническими знаниями в операционных системах Windows 8 и выше.

Назначение автоматизированного варианта решения задачи является использование голосового управления, для дополнительного инструмента взаимодействия с компьютером.

Программный продукт позволяет использовать полноценного голосового ассистента для ОС Windows. Это является устранением недостатка аналогичного продукта Cortana разработанного компанией Microsoft, адаптированного для русскоговорящих пользователей.

1.4.2 Общая характеристика организации решения задачи на ЭВМ

Задачи автоматизации требует написания программного продукта с нуля, то есть необходимо написать программу, которая будет записывать голос через микрофон, анализировать фразу пользователя и делать соответствующие действия.

Для анализа голосовых сообщений используется сервис компании Google SpechToText. Соответственно для разработки проекта требуется интеграция с этим сервисом и наличия ключа разработчика.

Программный продукт является клиент–серверным приложением с модульной архитектурой и имеющее внутреннюю базу данных. Архитектура клиент–сервер – это сетевая архитектура, в которой взаимодействуют

устройства, называемые клиентами и серверами. Данная архитектура может использоваться как для физических устройств, так и для программного обеспечения независимо от распределения логических компонентов приложения между клиентом и сервером.

Реализация этой архитектуры связана с использованием стороннего сервиса компании Google. Связь с этим сервисом осуществляется по протоколы HTTPS и является защищенным.

Клиентское приложение является модульным, эта архитектура необходима для создания оптимальной нагрузки на вычислительные мощности компьютера.

Внутренняя база данных необходима для хранения информации пользователя. Для упрощения обработки информации и организации потоков данных, база данных находится в самом приложении. Это так же обеспечивает целостность данных при изменении ОС.

Такая архитектура приложения является оптимальной, так как распределяет нагрузку между клиентом и сервером. Использование другой архитектуры может усложнить бизнес–логику приложения и масштабирование. Использование более сложной архитектуры приложения повлечет сложное взаимодействие между модулями приложения, масштабирование приложения и внесение изменений в программный продукт.

1.4.3 Формализация расчётов

Важнейший процесс в работе программного продукта – это распознавание речи. Рассмотрим один их методов распознавания речи.

Распознавание речи включает в себя два основных этапа: предварительную обработку сигнала и его классификацию. На этапе предварительной обработки исходный сигнал преобразуется в векторы признаков, на основе которых затем будет произведена классификация. Этот этап может включать в себя следующие шаги:

- преобразование сигнала из аналоговой формы в цифровую;
- применение фильтров для подавления шумов;
- выделение границ речи;
- выделение признаков сигнала.

Наиболее распространенные методы выделения признаков — это метод мел–частотных кепстральных коэффициентов и метод кепстральных коэффициентов на основе линейного предсказания. Мел–частотные кепстральные коэффициенты (MFCC) [16].

Пример алгоритма вычисления коэффициентов представлен на рисунке 1.6 и включает в себя следующие шаги:

- исходный сигнал разбивается на фреймы. Их размер обычно составляет от 10 до 40 мс. Фреймы накладываются друг на друга;
- к каждому фрейму применяется быстрое преобразование Фурье;
- переход к мел–шкале.

Рисунок 1.6 – Пример алгоритма вычисления коэффициентов [16]

Мел – это психофизическая единица высоты звука. Человек лучше различает звуки низкой частоты, чем высокой. На мел–шкале равное изменение частоты в мелах соответствует равному изменению ощущения высоты тона. То есть человек определит звук с частотой в 1000 мел в два раза “ниже”, чем 2000

мел, но для звуков частотой в 1000 гц и 2000 герц нет. Перевод герц в мелы происходит по формуле (4.1) [16].

$$m = 1124\ln\left(1 + \frac{f}{700}\right), \quad (4.1)$$

где m– частота, измерения в мелах;

f – частота, измерения в герцах.

Отображение сигнала на мел–шкалу происходит с помощью блока треугольных фильтров. На рисунке 1.7 представлены мел–фильтры.

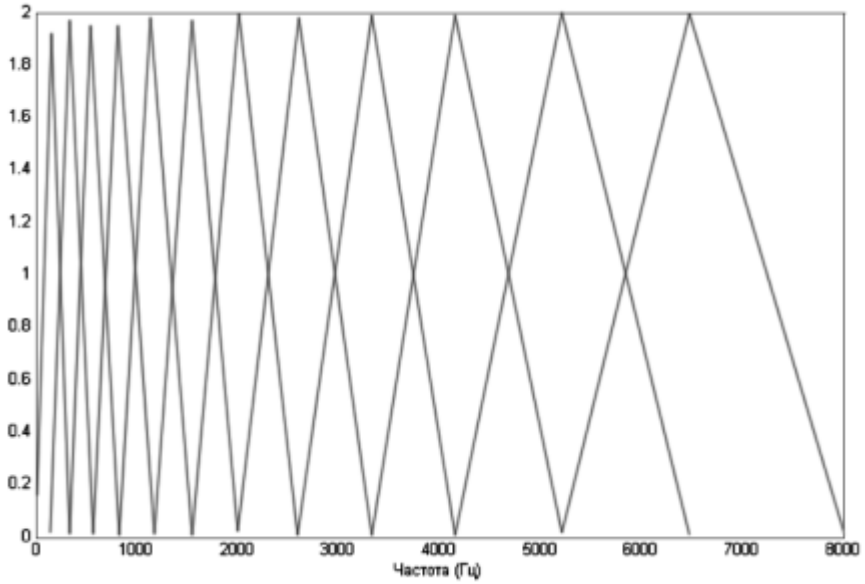


Рисунок 1.7 – Мел–фильтры [16]

Набор полученных на предыдущем шаге значений логарифмируется. Для получения кепстральных характеристик применяется дискретное косинусное преобразование.

Мел–фильтры – это кепстральные коэффициенты на основе линейного предсказания (LPCC). Линейное предсказание речи основано на линейной модели речеобразования, разработанной Фантом в 50–х годах XX века.

Проблему линейного предсказания можно сформулировать так: по значениям набора данных надо предсказать значение данных в последующей точке.

Пусть s(i)– анализируемый цифровой сигнал. При линейном предсказании оценка текущего отсчёта сигнала $s^{(i)}$формируется как линейная комбинация предшествующих отсчётов, рассчитанных по формуле (4.2) [16].

$$y(t) = \sum_{k=1}^{p} ay(i-k), \qquad (4.2)$$

где y(t)– коэффициент линейного предсказания;

a– анализируемый цифровой сигнал;

y – косинусное преобразование;

k – коэффициент смещения.

Задача линейного предсказания состоит в том, чтобы найти такой набор коэффициентов {ak}, для которого средний квадрат ошибки $(y(t) - y(t))2$минимален.

Коэффициенты линейного предсказания можно получить решением системы уравнений.

После получения набора признаков необходимо на их основе определить, к что за звук или слово находилось в исходном сигнале. Наиболее распространенные методы – это скрытые марковские модели и нейронные сети. Скрытые марковские модели Скрытая марковская модель – это модель из N скрытых состояний X и M наблюдаемых значений Y, определяется как тройка и вычисляется по формуле (4.3) [16].

$$\lambda = (A, B, \pi), \qquad (4.3)$$

где A – матрица вероятностей переходов между состояниями;

B– матрица вероятностей наблюдений выходных значений;

π – вектор вероятностей начальных состояний.

Пример диаграммы переходов в скрытой марковской модели. В обычной марковской модели состояние видимо наблюдателю, поэтому вероятности переходов — единственный параметр. В скрытой марковской модели можно наблюдать только переменные, на которые оказывает влияние данное состояние [16].

1.5 Анализ существующих разработок, выбор и обоснование стратегии автоматизации и способа приобретения. Обоснование выбора технологии проектирования

На сегодняшний день существует несколько голосовых помощников. Рассмотрим наиболее популярные из них:

- Яндекс Алиса;
- Amazon Alexa;
- Microsoft Cortana;
- Siri;
- Google Assistant.

«Яндекс Алиса» – Продукт компании Яндекс. «Алиса» – не искусственный интеллект (ИИ), который человечество до сих пор не создало. Да и в ближайшем будущем вряд ли создаст, ведь для разработки ИИ нужно хотя бы понять, как работает человеческий мозг. Поэтому на данный момент все голосовые помощники – лишь имитация сознания.

«Алиса» использует дедуктивное обучение, формализацию данных от полученных кодов, введённых техническими специалистами в созданную эмпирическую базу данных. От того, она всегда любит симулировать сарказм, иронию, потребности, заинтересованность. Имеет определённые правила заполнения шаблонов, выдачу наиболее релевантных ответов [5].

«Алиса» распознаёт и синтезирует речь с помощью платформы SpeechKit, которую в Яндексе начали разрабатывать ещё с 2012 года. Как утверждают компания, скорость распознавания синтеза речи составляет 1,1 секунды.Software development kit(SDK) продолжает расширяться и сейчас, сторонним разработчикам доступна возможность реализовывать на полноценном уровне обучение по нейронным сетям [5].

Amazon Alexa – голосовой помощник компании Amazon. Идея Alexa была простой и понятной, это голосовой помощник, которого можно разбудить, обратившись к нему по имени. Выбор имени Alexa не случаен, в английском языке четко распознается звук X, это уловка, чтобы устройство не ошибалось и реагировало на одно слово, а не на два, как это сделано у Google с его фразой «OK, Google». Помощник после обращения к нему умеет сообщать нужную информацию, например, рассказывать о погоде, пробках, зачитывать новости или даже отправлять письма другим людям, которые вы тут же диктуете. Позднее Alexa научилась управлять элементами умного дома, например, термостатами от Nest, лампочками Hue от Philips и другими. Вы просто говорите, что нужно изменить температуру, и Alexa отправляет эту команду на нужные устройства [11].

Также как в Siri, в Alexa можно вести диалоги и задавать вопросы, многие ответы шутливы, на многие вопросы система ищет ответы в интернете. Появлением Alexa и других подобных систем мы обязаны прогрессу в двух областях – распознавании речи и распространении дешевого интернета, когда каждая квартира постоянно подключена к сети. Прежде чем рассмотреть устройство Alexa и ей подобных, сделаю ремарку в отношении Siri, Google Assistant и других голосовых помощников на смартфонах и планшетах. Технология, которая лежит в основе этих сервисов, полностью идентична и никак не отличается [11].

Microsoft Cortana – разработка Кортаны велась в течение долгих лет отдельно созданным подразделением Microsoft, которое занималось обучением

ассистента и его внедрением в мобильные продукты различных операционных систем. Релизная версия Cortana была представлена 2 апреля 2014 года на мероприятии Build для разработчиков в Сан–Франциско [12].

Произношение Кортаны имеет максимально приближенную речь к человеческому диалекту. К этому стремятся многие компании, которые занимаются разработкой ассистентов на технологии искусственного интеллекта и машинного обучения. Кроме того, Microsoft Cortana является первым и единственным помощником, который имеет ряд эмоциональных состояний. Ассистент может быть спокойным, услужливым, задумчивым, веселым, оптимистичным, бдительным и восторженным. Эти эмоции Cortana показывает при помощи кнопки вызова, которая меняет индикацию в зависимости от настроения виртуальной программы [12].

Siri – это самообучающаяся система искусственного интеллекта, в конечно итоге дающая пользователю ощущение взаимодействия с умным виртуальным помощников. Составляющие системы Siri не являются абсолютным новшеством, но в сочетании скрыты невиданные раньше возможности [11].

Siri имеет возможность управлять компьютером, но только под управлением Mac OS. Компьютеры с данной ОС являются дорогостоящими и не пользуются популярностью в России и СНГ.

Для России и СНГ наиболее популярной системой является Алиса и Google Assistant. Но эти продукты не являются универсальными решениями и имеют скудный функционал работы с операционной системы [11].

Google Assistant спроектирован для операционной системы Android и использует тот же инструментарий, что и разрабатываемый программный продукт. Минусом Google Assistant является то, что он совершенно не умеет работать с операционной системой Windows и является голосовым помощником только для мобильных устройств [11].

Все перечисленные программные продукты не позволяют управлять операционной системой Windows 8 и выше при помощи голоса. Поэтому существует потребность в разработке программного продукта голосового управления компьютером для операционной системы Windows 8 и выше.

Разработанный программный продукт позволяет при помощи голоса:

- открывать системные приложения;
- искать информацию в сети Интернет;
- возможность использования технических и информационных мощностей компьютера;
- выключать компьютер;
- перегружать компьютер;
- перевести компьютер в режим гибернации.

Разработанный программный продукт является актуальным решением для пользователей в настоящее время, потому что позволяет людям с ограниченными возможностями управлять компьютером с операционной системой Windows 8 и выше.

1.6 Обоснование проектных решений по видам обеспечения

1.6.1 Обоснование проектных решений по техническому обеспечению

Техническое обеспечение (ТО) — совокупность средств реализации управляющих воздействий, средств получения, ввода, подготовки, преобразования, обработки, хранения, регистрации, вывода, отображения, использования и передачи информации и эксплуатационной документации [16].

При выборе программного и аппаратного обеспечения, в качестве определяющих, были выбраны следующие критерии:

- низкие расходы на сопровождение и модификацию программного и аппаратного обеспечения;
- использование технологий с открытым исходным кодом;

- простота в использовании.

1) Характеристики ПК для разработчика:

- процессор Intel Core–i5;
- жесткий диск Seagate 1024 Gb;
- оперативная память DDR3 4Gb;
- видеокарта Geforce8800;
- операционная система Microsoft Windows 10 Professional;

2) Характеристики ПК сервера:

- процессор Intel Core–i3;
- жесткий диск Seagate 2500 Gb;
- оперативная память DDR3 16Gb;
- видеокарта Geforce 6600;
- операционная система Microsoft Windows Server 2008;

3) Характеристики ПК рабочие станции:

- процессор Intel Core–i3;
- жесткий диск Seagate 500 Gb;
- оперативная память DDR3 2Gb;
- видеокарта Geforce 6600;
- операционная система Microsoft Windows 10 Professional;
- принтеры, ксероксы, сканеры,
- МФУ HPMP–1020dn;
- принтер HPP1505.

Чтобы комфортно эксплуатировать программу требуется, чтобы компьютер был сконфигурирован следующим образом:

- процессор: Intel Core–i3;
- жесткий диск Seagate 500 Gb;
- оперативная память DDR3 2Gb;
- видеокарта Geforce 6600;
- операционная система Microsoft Windows 10 Professional.

1.6.2 Обоснование проектных решений по информационному обеспечению

Информационное обеспечение (ИО) — совокупность единой системы классификации и кодирования информации, унифицированных систем документации, схем информационных потоков, циркулирующих в организации, а также методология построения баз данных [16].

Программный продукт имеет внутримашинное информационное обеспечение.

Внутримашинное информационное обеспечение представляет собой специальным образом организованные данные, которые подлежат автоматизированному хранению, накоплению, обработке, поиску и передаче в удобном для восприятия техническими средствами виде [17].

Программный продукт имеет базу данных с таблицами, в которых хранится информация о пользовательских командах, использующихся в приложении. Локальная база данных необходима для хранения информации пользователя, например, пользовательских команд запуска, и системной информации. Наличие локальной базы данных открывает возможность распределения информационной нагрузки в программном продукте и возможность сохранять пользовательские настройки.

Программный продукт хранит большую часть информации в оперативной памяти компьютера, это необходимо для нормально функционирования приложения. В нем хранится временная информация о системных процессах. Без оперативной памяти программный продукт не сможет функционировать в штатном режиме [17].

Программный продукт хранит журнал ошибок, внутри файловой системы. Эти файлы нужны для отладки приложения. Хранение журнала ошибок очень важный элемент, так как он позволяет эффективно отлаживать программный код и следить за системными ошибками, которые могут возникнуть в ходе работы приложения.

К программному продукту необходима документация пользования, предоставляющая инструкцию по установке и настройке приложения на компьютер. Документация важна для пользователя. С помощью нее пользователь сможет настроить программный продукт по своему желанию, не обращаясь за помощью к администраторам и разработчикам.

1.6.3 Обоснование проектных решений по программному обеспечению

Программное обеспечение (ПО) - программа или множество программ, используемых для управления компьютером [16].

Для разработки дипломного проекта была выбрана Visual Studio 2013.

Visual Studio 2013 – линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, а также веб–сайты, веб–приложения, веб–службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework и Silverlight.

Visual Studio включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб–редактор, дизайнер классов и дизайнер схемы базы данных. Visual Studio позволяет создавать и подключать сторонние дополнения (плагины) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода (как, например, Subversion и Visual Source Safe), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода на предметно–ориентированных языках программирования) или инструментов для прочих аспектов процесса разработки программного обеспечения (например, клиент Team Explorer для работы с Team Foundation Server).

C# - объектно–ориентированный язык программирования. Разработан в 1998—2001 годах группой инженеров под руководством Андерса Хейлсберга в

компании Microsoft как язык разработки приложений для платформы Microsoft .NET Framework и впоследствии был стандартизирован как ECMA–334 и ISO/IEC 23270.

C# относится к семье языков с C–подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML.

Переняв многое от своих предшественников — языков C++, Pascal, Модула, Smalltalk и, в особенности, Java — C#, опираясь на практику их использования, исключает некоторые модели, зарекомендовавшие себя как проблематичные при разработке программных систем, например, C# в отличие от C++ не поддерживает множественное наследование классов (между тем допускается множественное наследование интерфейсов) [25].

1.6.3 Обоснование проектных решений по технологическому обеспечению

Технологический процесс обработки информации — это определенный комплекс операций, выполняемых в строго определенной регламентом последовательности с использованием определенных методов и инструментальных средств обработки, охватывающих все этапы обработки данных, начиная с процесса регистрации первичных данных и заканчивая передачей результатной информации пользователю для выполнения функций управления [3].

Технологический процесс можно разделить на следующие этапы:

1) первичный;
2) подготовительный;
3) основной;

4) заключительный;

На первичном этапе осуществляется процесс сбора исходных данных, их регистрация и передача для ввода в ПК.

Подготовительный этап охватывает операции по приему, контролю и регистрации входной информации, и переносу ее на ПК.

На основном этапе осуществляется обработка информации на ПК.

На заключительном этапе осуществляется контроль, выпуск и передача результатной информации пользователю.

Ввиду специфики программного продукта, в технологический процесс для задачи выпускной квалификационной работы входит только 3 и 4 этапы.

2 ПРОЕКТНАЯ ЧАСТЬ

2.1 Информационное обеспечение задачи

2.1.1 Информационная модель и ее описание

Информационная модель имеет вид структурного представления движения информационных потоков с момента поступления входной информации до момента выдачи выходных форм.

Информационная модель должна соответствовать следующим требованиям:

- обеспечивать отображение предметной области и давать возможность получить интегрированное представление о предметной области;
- содержать информацию о предметной области, достаточную для дальнейшего проектирования.

Информационная модель задачи, представленная на рисунке 2.1, включает в себя следующие объекты:

- входные документы и формы (аудиофайл голосовой команды, форма добавления программы);
- выходные документы, которые формируются на основании полученных данных (сообщение распознавании команды; сообщение что программа не найдена, результат распознавания речи).

-

Рисунок 2.1 – Информационная модель задачи

2.1.2 Характеристика первичных документов с нормативно— справочной и входной оперативной информацией

Входная информация представляет собой описание состава входных документов, соответствующих им экранных форм размещения данных.

Голосовая команда — это аудио файл записи голосовой команды в формате FLAC.

FLAC — свободный кодек, предназначенный для сжатия аудиоданных без потерь. FLAC является аудиофайлом, сжатие которого было произведено без потери качества в звучании. Формат файла с расширением .flac считается ниже формата MP3 со стороны показателей степени сжатия, однако формат FLAC однозначно демонстрирует лучшие результаты качества звука.

Использование аудиофайлов в формате FLAC обусловлено требованием библиотека распознавания голоса.

Форма настроек — форма, на которой представлено ключевое слово, кнопка выбора программы. Эта форма позволяет создать синоним для программы, тем самым создать голосовую команду запуска программы.

2.1.3 Характеристика базы данных

При создании данного программного продукта было использовано пространство имен System.Data.OleDb, которое представляет собой набор классов, используемых для доступа к источникам данных OLE DB в управляемом пространстве. Данное пространство имен использовалось для работы с СУБД Microsoft Office Access.

Microsoft Office Access — реляционная СУБД корпорации Microsoft. Имеет широкий спектр функций, включая связанные запросы, связь с внешними таблицами и базами данных.

Система Access – это набор инструментов конечного пользователя для управления базами данных. В ее состав входят конструкторы таблиц, форм, запросов, отчетов и страниц доступа к данным. Эту систему можно рассматривать и как среду разработки приложений. Используя макросы или модули для автоматизации решения задач, можно создавать ориентированные на пользователя приложения такими же мощными, как и приложения, написанные непосредственно на языках программирования. При этом они будут включать кнопки, меню и диалоговые окна [20].

2.1.4 Характеристика результатной информации

Результатная информация представляет собой описание состояния выходных документов, соответствующих им экранных форм размещения данных.

Сообщение о распознавании голосовой команды, позволяет пользователю проверить насколько точно программа смогла распознать голосовую команду.

При использовании ключевых слов программа выполняет действие, соответствующее этому ключевому слову.

2.2 Программное обеспечение задачи

2.2.1 Общие положения

После загрузки приложения пользователь попадает на главную форму. Вверху формы находится кнопки «Настройки» и «Запись». Кнопка «Запись» записывает голос пользователя и распознает голосовую команду.

Кнопка «Настройки» открывает диалоговое окно добавления голосовой команды. Окно содержит форму со следующими полями:

- текстовое поле «Ключевое слово»;
- кнопка «Добавить путь»;
- кнопка «Добавить программу».

Форма «Настройки» позволяет создать собственную голосовую команду для запуска программы. Для создания команды пользователь должен ввести ключевое слово и указать путь к исполняемому файлу, после чего нажать кнопку «Добавить программу». Данные с формы попадут в базу данных и программа сможет открыть пользовательскую программу по ключевому слову.

Форма «Настройки» представлена в пункте 2.4.

2.2.2. Структурная схема программной системы

Диаграмма вариантов использования представленная на рисунке 2.3. Пользователь программного продукта может:

- создавать пользовательские команды запуска программ;
- открывать пользовательские программы при помощи голосовой команды;
- выключать компьютер при помощи голосовой команды;
- перезагружать компьютер при помощи голосовой команды;
- включать режим гибернации при помощи голосовой команды;

- осуществлять поиск информации в сети интернет при помощи голосовой команды;

- сохранять информацию в оперативной памяти компьютера при помощи голосовой команды.

Рисунок 2.3 – Диаграмма вариантов использования

Диаграмма активности представлена на рисунке 2.4.

Диаграмма активности представлена для одной итерации запуска программного продукта.

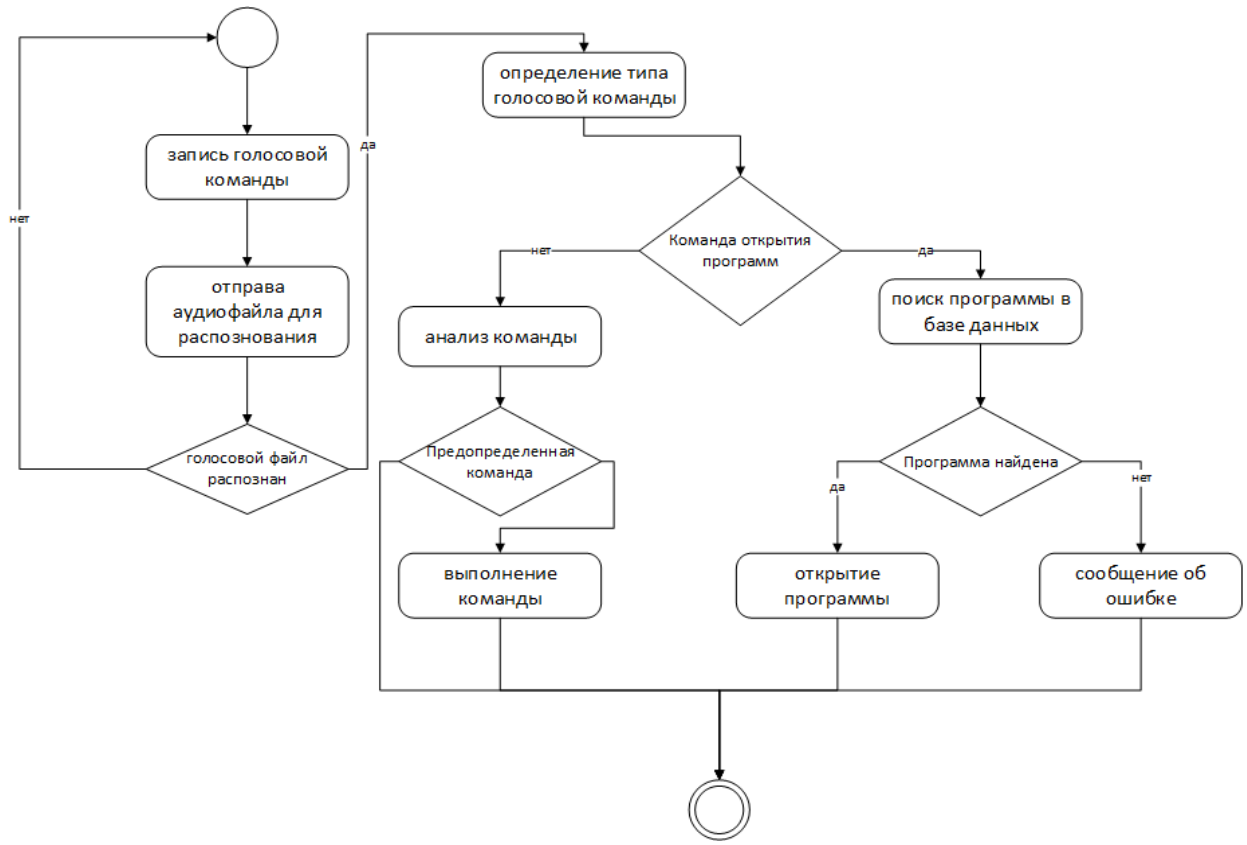


Рисунок 2.4 – Диаграмма активности

2.2.3 Описание программных модулей и библиотек

Структура программной системы, описывающей используемые библиотеки и компоненты, представлена таблице 2.1.

Таблица 2.1 Структура программной системы

Название	Описание
NAudio	Библиотека записи звука
FlacLibSharp	Библиотека работы с аудиофайлами формата flac
SpechToText	Библиотека для распознавания речи
.NET framework	Фреймворк для языка программирования C#
WPF forms	Библиотека для создания интерфейсов

LinqToXml	Библиотека для работы с XML
LinqToJson	Библиотека для работы с JSON
Windows.Forms	Компонент для работы с формами

2.3 Технологическое обеспечение задачи

2.3.1 Организация технологии сбора, передачи, обработки и представления информации

Технологический процесс — это процесс технологических операций, необходимых для выполнения определённого вида работ. В данном случае под работой понимается процесс создания программного продукта голосового управления компьютером [25].

Создание программного продукта голосового управления компьютером представляет собой сложный технологический процесс, который требует больших затрат времени и труда. Для реализации проекта весь процесс разбивается на составные части — этапы:

1. Анализ задач, которые должны быть решены в рамках проекта. Проводится анализ существующих программных продуктов, конкурентов для выявления уже существующих решений, анализ их достоинств и недостатков. На основе этого предлагается решение поставленной задачи.
2. Анализ инструментов, которые способны реализовать задачу программного продукта. Проводится анализ технологий и инструментов для распознавания голоса, для выявления наиболее удобных.
3. Создание дизайна приложения.
4. Создание базовой архитектуры приложения. Производится создание базовых классов и модулей приложения, для дальнейшего использования.

5. Создается графический интерфейс.

Разработка функциональных возможностей программного продукта.

2.4 Описание контрольного примера реализации проекта

Для запуска приложения необходимо зайти в папку Debug и открыть файл «Speak.exe». На рисунке 2.5 представлена форма главного окна приложения.

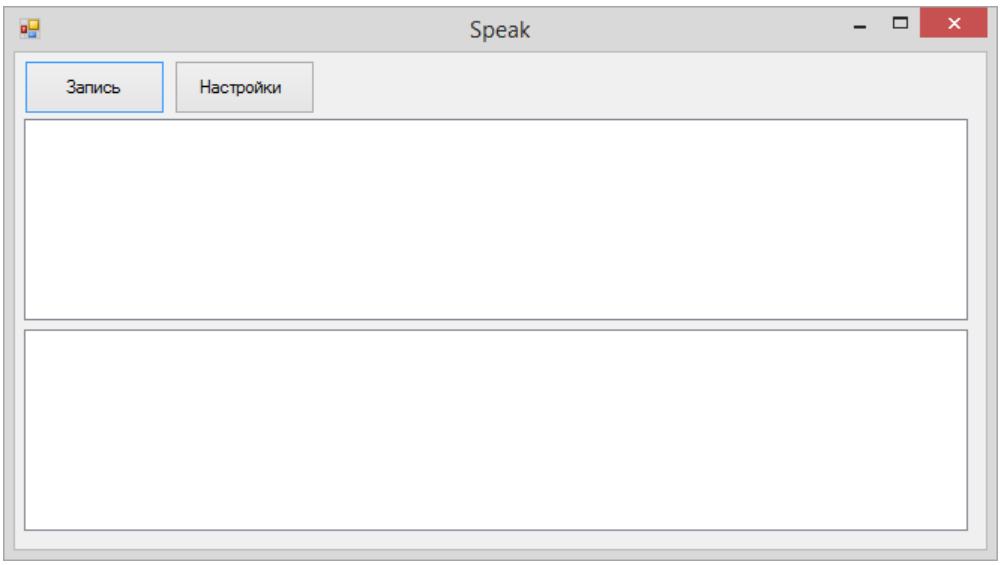


Рисунок 2.5 — Форма главного окна приложения

При нажатии на кнопку «Настройки» открывается форма настроек приложения. На рисунке 2.6 представлена форма настройки приложения.

На главной форме приложения расположено 2 кнопки «Запись» и «Настройки», журнал поступающих ответов, журнал обработанных результатов.

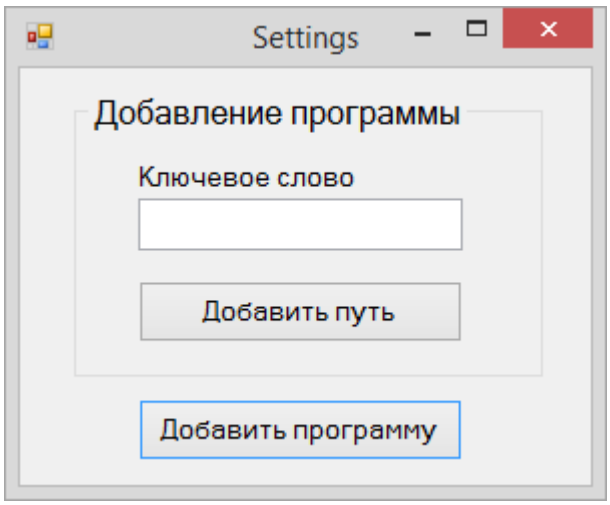


Рисунок 2.6 — Форма настройки приложения

На форме настроек приложения можно добавить программу в базу данных приложения. Для этого необходимо ввести ключевое слово, которое будет ассоциироваться с этим приложением и путь к exe—файлу.

На рисунке 2.7 представлен пример заполненной формы настроек.

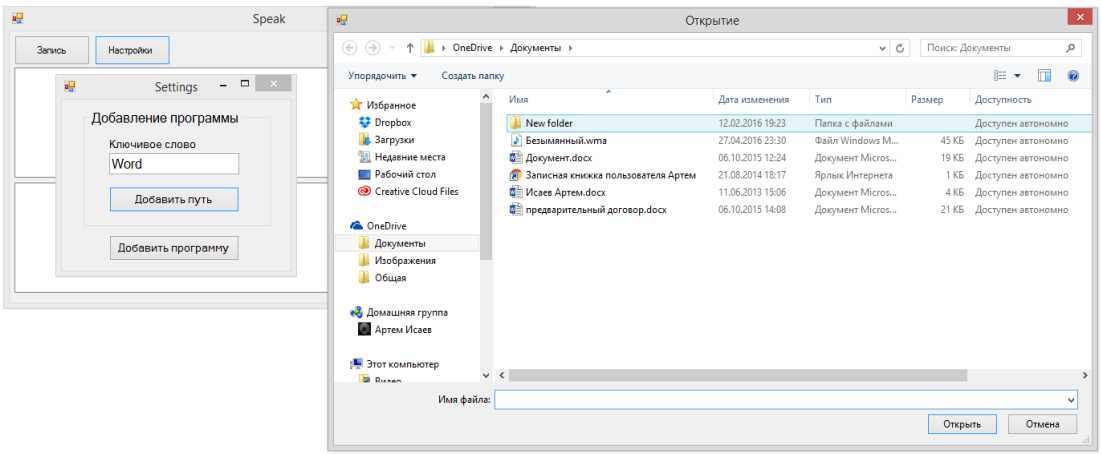


Рисунок 2.7 — Пример заполненной формы настроек

После нажатия на кнопку «Добавить программу» создается запись в базе данных, соответствующая введенным значениям.

Для записи голосовой команды необходимо нажать на кнопку «Запись» на главной форме. Текст кнопки изменится на «Стоп» и начнется запись голоса.

При повторном нажатии на кнопку «Стоп» произойдет остановка записи звука и отправка аудиофайла на обработку в дата-центр Google. При успешном ответе строка состояния сообщит о том, что запрос успешно выполнен. На рисунке 2.8 представлен пример успешного запроса и обработки аудиофайла

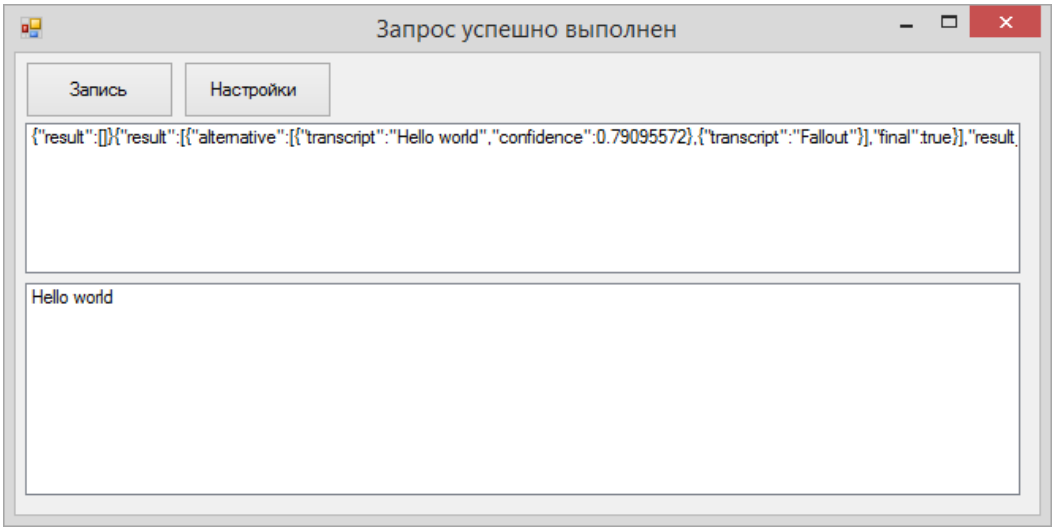


Рисунок 2.8 — Пример успешного запроса и обработки аудиофайла

При успешном выполнении запроса в правом верхнем углу экрана появляется Push -уведомление с переведенным сообщением пользователя. На рисунке 2.9 представлено Push - уведомление с переведенным сообщением пользователя

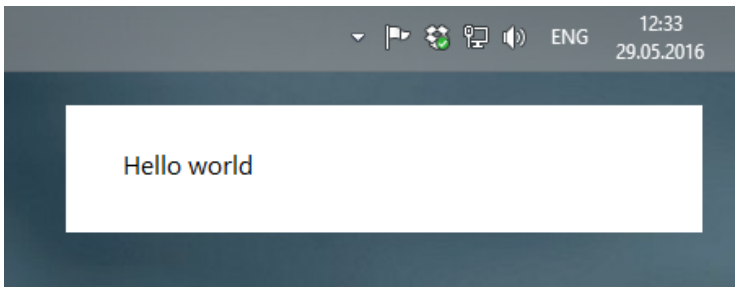


Рисунок 2.9 — Уведомление с переведенным сообщением пользователя

При успешном запросе в журнале пришедших ответов отображается ответ в формате JSON, полученный от Google. На рисунке 2.10 представлен

журнал поступающих ответов. А в журнале обработанных результатов отображается сказанное пользователем сообщение. На рисунке 2.11 представлен журнал обработанных результатов.

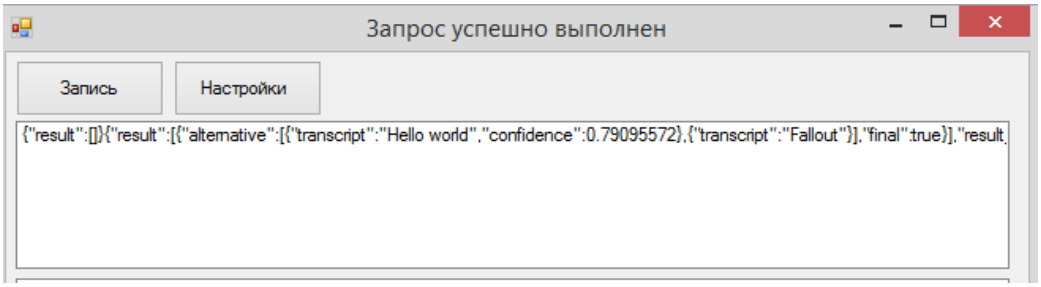


Рисунок 2.10 — Журнал поступающих ответов

Рисунок 2.11 — Журнал обработанных результатов

Для открытия программ необходимо произнести ключевое слово «Открыть» и после чего название программы, которую необходимо открыть. В случае, если данная программа отсутствует в базе данных программы выводится сообщение, говорящее это. На рисунке 2.12 представлено сообщение отсутствия программы в базе данных.

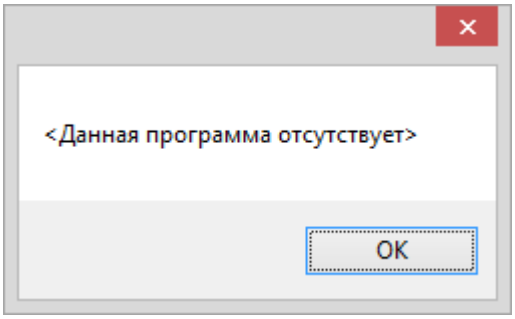


Рисунок 2.12 — Сообщение отсутствия программы в базе данных

В программе имеется возможность работать с браузером. Для этого необходимо сказать: «Открыть браузер», в этом случае откроется пустое окно браузера.

Если сказать: «Открыть браузер Яндекс блог», тогда окно браузера откроется на странице поиска с введенным запросом «Яндекс блог».

Имеется возможность записи текста в буфер обмена, для этого нужно сказать ключевое слово «Запомнить» и программа запомнит все, что идет после этого слова.

Программный продукт имеет такие ключевые слова как:

- выключить компьютер;
- перезагрузить компьютер;
- спящий режим.

Эти команды изменяют состояние компьютера.

3 ОБОСНОВАНИЕ ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА

3.1 Эффективность применения информационных технологий

Эффективность – это продуктивность использования ресурсов в достижении какой-либо цели.

Экономическая эффективность — это соотношение полезного результата и затрат факторов производственного процесса [1].

В основе описания экономической эффективности лежит сопоставление существующего и внедряемого технологических процессов (базового и проектного вариантов) на основе анализа затрат, необходимых для выполнения всех операций технологического процесса. Для обоснования экономической эффективности проекта может быть использована методика расчета прямой эффекта от внедрения информационной системы по сравнению с базовым вариантом существующей организации обработки информации.

Основным принципом расчёта экономической эффективности является сравнение двух вариантов обработки информации, например, ручной обработки данных и обработки данных с использованием ЭВМ, локальное использование программного средства или сетевое и т.п. Необходимо рассчитать затраты на разработку проекта.

Выводы об экономической эффективности делаются на основе вычисленных экономических показателей.

3.2 Расчет показателей экономической эффективности проекта

Оценка этапов разработки представлена в таблице 3.1.

Таблица 3.1 – Оценка этапов разработки программного продукта

Этап разработки	Трудоемкость в часах
Анализ задач, которые должны быть решены в рамках проекта	16

Анализ инструментов, которые способны реализовать задачу	16
Создание дизайна приложения	8
Создание базовой архитектуры приложения	32
Создается графический интерфейс	16
Разработка функциональных возможностей	40
Всего	128 часов

Коэффициент сложности задачи c - характеризует относительную сложность программы по отношению к так называемой типовой задаче, сложность которой принята равной единице (величина c лежит в пределах от 0,5 до 1,5). Для данного программного продукта сложность задачи берем 1,1 т.к. данный программный продукт не является сложным.

c = 1,1 – (коэффициент сложности программы)

Коэффициент увеличения затрат труда b, вследствие недостаточного описания задачи, в зависимости от сложности задачи принимается от 1 до 1,5. В связи с тем, что данная задача, потребовала уточнения и больших доработок, примем b = 1,4.

Коэффициент квалификации разработчика k определяется в зависимости от стажа работы и составляет:

- для работающих до двух лет - 0,8;
- от трех лет до пяти лет - 1,0;
- от пяти до восьми - 1,2;
- более восьми лет -1,5.

Поскольку стажа работы по специальности у разработчика составляет 3,5 года, возьмем k= 1,0.

Оклад программиста фирмы равен 25000 рублей (З/П). С учётом

использования коэффициентов заработной платы основная заработная плата разработчика программного продукта составит:

З/П осн. = оклад *c*b*k = 25000 руб. * 1,1 * 1,4*1,0 =38500 руб. в месяц.

За время разработки программного продукта дополнительная заработная плата не выплачивалась.

Страховые взносы берутся в размере 30% от основной и дополнительной заработной платы:

ФО=З/п осн *30%=38500 *30%=11550р. в месяц.

Итоговая заработная плата за весь период разработки программного продукта (128 ч.) составит:

З/П общ = З/П осн + страховые взносы*128 / 176ч = 38500 + 11550 * 128 / 176 = 46900 за весь период разработки программного продукта.

Величина оплаты труда сотрудника составит:

ЗП сум. = З/П осн+ Страховые взносы = 38500 + 11550 = 50050 руб.

Содержание и эксплуатация вычислительной техники:

Свт = См-ч * Число часов отладки,

где См-ч – стоимость машино-часа.

Для расчета часов отладки суммируем время:

технический проект + рабочий проект + документация и внедрение.

Число часов отладки = 45час. + 75 час. + 50 час. = 170 час.

Стоимость машино-часа рассчитывается, как сумма составляющих:

(Ст-ть_эл_эн_в_год+Аморт в_год+Затраты_на_ремонт_за_год)/Фвт,

где Фвт - действительный фонд времени работы вычислительного комплекса.

Стоимость 1 КВТ/час электроэнергии составляет: 4,5 руб.

Один компьютер потребляет в среднем 300 Вт в час.

За год оплата за электроэнергию, потребляемую одной ПЭВМ, составляет:

8 ч. * 22 дня * 12 мес. * 0,3 КВт/ч. *4,5руб. = 2851,2 руб.

Программный продукт разрабатывался на компьютере, который является собственностью предприятия. Так как до этого он уже использовался по назначению в течение более 4-х лет, мы не будем производить расчет амортизации. Срок полезного использования компьютера составляет 4 года.

ПК был приобретен в декабре месяце 2015 года. Стоимость компьютера составляет 22500 руб.

Стоимость ремонта ПЭВМ= 22500 * 5% = 1125 руб.

В год компьютер работает 2 112 часов, на его ремонт уходит 5% времени, значит, окончательное время работы компьютера 2 006 часов в год.

Стоимость машинного часа равна:

(2851,2 + 1125) / 2006 = 1,98 руб.

Содержание и эксплуатация вычислительного комплекса на время разработки программного продукта составляет:

170 час. * 1,98 руб. = 3366 руб.

Легальная версия MS Office уже была установлена на компьютере, следовательно, дополнительные затраты на программное обеспечение не требуются.

Затраты на разработку программного продукта приведены в таблице 3.2.

Таблица 3.2 - Затраты на разработку программного продукта

№	Наименование расходов	Затраты, руб.
1	Общая заработная плата	38500 руб.
2	Страховые взносы	11550 руб.
3	Содержание и эксплуатация вычислительного комплекса.	3366 руб.
4	Программное обеспечение	0 руб.
ИТОГО:		53416 руб.

Поскольку программный продукт является благотворительным, экономическая выгода не была основной задачей.

Для компании разработка программного продукта убыточна, так как стоимость разработки составило 53416 руб.

Программный продукт распространяется бесплатно, что является выгодным для пользователей. Поэтому для пользователей программный продукт является экономически выгодным.

Заключение

В результате выполнения выпускной квалификационной работы разработан программный продукт, позволяющий управлять персональным компьютером с помощью голоса.

Одним из успешных показателей проделанной работы является точность распознавания голосовой команды.

Показатель точности распознавания голосовой команды был достигнут не только благодаря применению технологии распознавания речи, но и алгоритму программы, который позволяет включать и выключать запись голосовых команд, когда это потребуется. Тем самым снижая вероятность неправильного распознавания речи.

В результате выполнения выпускной квалификационной работы приобретены практические навыки не только в области программирования, но и в области экономики, менеджмента.

Разработанная программа устойчиво выполняет все свои функции:

- записи голоса пользователя;
- распознавание речи пользователя;
- открытия программ MS Office;
- открытия стандартных программ Windows;
- открытия браузера;
- открытия браузера на заданной странице;
- выключения компьютера;
- перезагрузки компьютера;
- запоминания фраз в буфер обмена;
- добавления новых программ в базу данных;
- гибернации компьютера.

Библиографический список

1. Агуров П.В. C#. Разработка компонентов в MS Visual Studio. – СПб.: БХВ–Петербург, 2019 – 327 с.

2. Буч Г. Язык UML. Руководство пользователя: Пер. с англ. – М.: ДМК, 2017 – 496 с.

3. Воробьева С. А. Методы распознавания речи [Электронный ресурс]. URL https://moluch.ru/archive/130/36213/ (дата обращения: 08.01.2020)

4. Дениел Х. Microsoft Corporation Принципы проектирования программного обеспечения. Учебный курс, MCSD. – М.: “Русская редакция”, 2018. – 127 с.

5. Как это работает? Распознавание речи [Электронный ресурс]. URL: https://yandex.ru/blog/ (дата обращения: 16.12.2019)

6. Козленко Л. Проектирование информационных систем [Электронный ресурс]. URL: http://www.interface.ru (дата обращения: 21.11.2019)

7. Компонентно–ориентированное программирование [Электронный ресурс]. URL: http://obertone.ru/ (дата обращения: 16.12.2019);

8. Нейгел К. C# 2018 для профессионалов / К. Нейгел. – М.: Вильямс, 2018. – 195 с.

9. Научная библиотека [Электронный ресурс]. URL: http://www.frolov–lib.ru/ (дата обращения: 14.11.2019)

10. Орлов С. Технологии разработки программного обеспечения. Учебник. – СПб.: Питер, 2017 – 241 с.

11. Особенности C# [Электронный ресурс]. URL: https://books.google.ru (дата обращения: 03.12.2019)

12. Официальный блог компании Google [Электронный ресурс]. URL: http://googlerussiablog.blogspot.ru/ (дата обращения: 20.12.2019)

13. Официальный блог MSDN компании Microsoft [Электронный ресурс]. URL: https://blogs.msdn.microsoft.com/ (дата обращения: 22.12.2019)

14. Официальный сайт компании Microsoft [Электронный ресурс]. URL: https://www.microsoft.com (дата обращения: 27.12.2019)

15. Подходы к выделению речи из исходного сигнала для системы обработки речи [Электронный ресурс]. URL: http://moluch.ru/ (дата обращения: 10.12.2019)

16. Робинсон С. C# для профессионалов. – М.: ЛОРИ, 2017. – 812 с.

17. Сайт разработчиков MySQL [Электронный ресурс]. URL: http://www.mysql.com/ (дата обращения: 08.01.2019)

18. Сайт разработчиков PHP [Электронный ресурс]. URL: http://www.php.net/ (дата обращения: 10.01.2019)

19. Сайт компании JetBrains [Электронный ресурс]. URL: https://jetbrains.ru/ (дата обращения: 13.01.2019)

20. Сайт компании Microsoft [Электронный ресурс]. URL: https://visualstudio.microsoft.com/com/ (дата обращения: 13.01.2019)

21. Сайт компании Adobe [Электронный ресурс]. URL: https://www.adobe.com/ru/ (дата обращения: 13.01.2019)

22. Синтез и распознавание речи. Современные решения [Электронный ресурс]. URL: sernam.ru (дата обращения: 14.12.2019)

23. Структура системы управления базами данных [Электронный ресурс]. URL: http://edu.dvgups.ru/ (дата обращения: 28.11.2019)

24. Уотсон К. Visual C# 2018. Базовый курс. – М.: Вильямс, 2018. – 314 с.

25. Фуксман А.Л. Технологические основы создания программных систем. – М.: Статистика, 2017. – 184 с.

26. Эрик дж. Браунде. Технология разработки программного обеспечения. – СПб.: Питер 2019. – 655 с.

Приложения

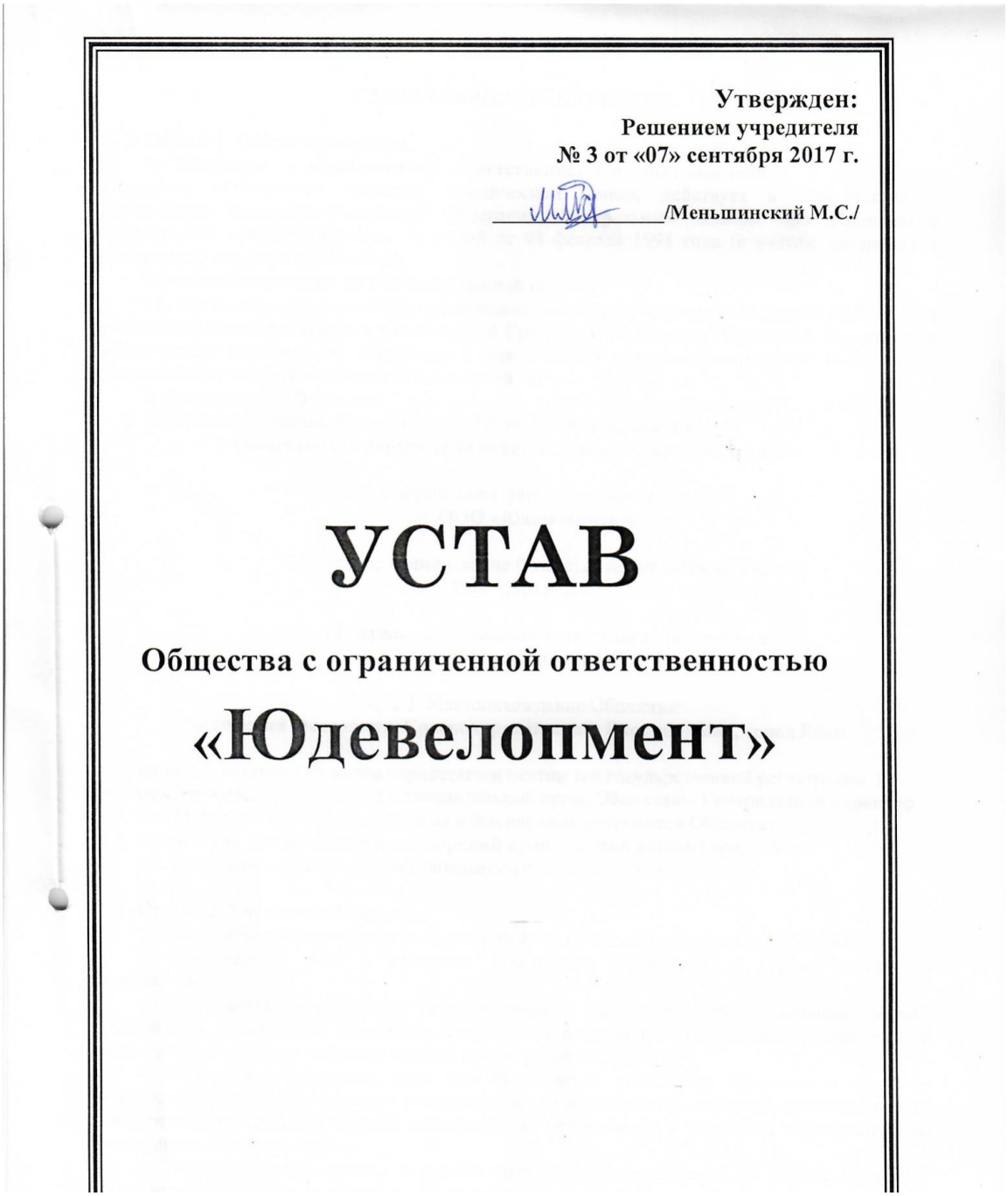
(с. 57 — 95)

Приложение А

(справочное)

Учредительные документы предприятия

(с. 58 — 62)



Утвержден:
Решением учредителя
№ 3 от «07» сентября 2017 г.

_____/Меньшинский М.С./

УСТАВ

Общества с ограниченной ответственностью

«Юдевелопмент»

Рисунок А.1 - Титульный лист устав предприятия

ГЛАВА I. ОБЩИЕ ПОЛОЖЕНИЯ.

Статья 1. Общие положения.

1. **Общество с ограниченной ответственностью «Юдевелопмент»**, в дальнейшем именуемое **«Общество»**, является юридическим лицом, действует в соответствии с Гражданским Кодексом Российской Федерации, Федеральным законом «Об обществах с ограниченной ответственностью» № 14-ФЗ от 08 февраля 1998 года (с учетом изменений и дополнений) и настоящим Уставом.

Общество учреждено на неограниченный срок.

1.1. Настоящая редакция Устава утверждена решением учредителя № 3 от 07.09.2017 года в соответствии с положениями части первой Гражданского Кодекса Российской Федерации и Федеральным законом «Об обществах с ограниченной ответственностью» № 14-ФЗ от 08 февраля 1998г. (с учетом изменений и дополнений).

2. Наименование Общества:

2.1. Полное фирменное наименование Общества на русском языке:
Общество с ограниченной ответственностью «Юдевелопмент»

2.2. Сокращенное фирменное наименование:
ООО «Юдевелопмент»

2.3. Полное наименование Общества на английском языке:
Udevelopment LLC

2.4. Сокращенное наименование на английском языке:
Udevelopment LLC

2.5. Местонахождение Общества:
Российская Федерация, Краснодарский край, Ейский район, город Ейск

Место нахождения Общества определяется местом его государственной регистрации. По указанному адресу размещается Исполнительный орган Общества – **Генеральный директор**.

2.6. Место хранения учредительных и финансовых документов Общества:
Российская Федерация, Краснодарский край, Ейский район, город Ейск

2.7. Почтовый адрес Общества совпадает с его местом нахождения.

Статья 2. Участники Общества.

1. Участниками Общества могут быть признающие положения настоящего Устава:

- юридические лица и граждане Российской Федерации и других государств Содружества;
- иностранные юридические лица, включая, в частности, любые компании, фирмы, предприятия, организации, ассоциации, созданные и правомочные осуществлять инвестиции в соответствии с законодательством страны своего местонахождения;
- иностранные граждане, лица без гражданства, российские граждане и граждане государств Содружества, имеющие постоянное место жительства за границей, при условии, что они зарегистрированы для ведения хозяйственной деятельности в стране их гражданства или постоянного местожительства.

2. Государственные органы и органы местного самоуправления не вправе выступать Участниками Общества, если иное не установлено Федеральным законом.

3. Порядок вступления Участников в Общество и выхода из него регулируется действующим законодательством Российской Федерации и положениями настоящего Устава.

4. Общество не может иметь в качестве [illegible]

Рисунок А.2 - Участники общества

Статья 3. Правовое положение Общества.

1. Общество считается созданным как юридическое лицо с момента его государственной регистрации в порядке, установленном федеральным законом о государственной регистрации юридических лиц.

2. Общество имеет в собственности обособленное имущество, учитываемое на самостоятельном балансе и отвечает по своим обязательствам этим имуществом.

Общество может от своего имени приобретать и осуществлять имущественные и личные неимущественные права, нести обязанности, быть истцом и ответчиком в суде.

3. Общество является коммерческой организацией с разделенным на доли Уставным капиталом и в качестве основной цели деятельности преследует извлечение прибыли в интересах Участников.

4. Общество может иметь гражданские права и нести гражданские обязанности с момента создания Общества, которым считается дата его государственной регистрации.

Общество вправе иметь гражданские права и нести гражданские обязанности, необходимые для осуществления любых видов деятельности, не запрещенных федеральными законами, если это не противоречит предмету и целям деятельности, ограниченным настоящим Уставом.

5. Правоспособность Общества прекращается в момент завершения его ликвидации, которым считается дата внесения записи о ликвидации Общества в единый государственный реестр юридических лиц.

6. Общество имеет круглую печать, содержащую его полное фирменное наименование на русском языке, с указанием ОГРН. Печать Общества может содержать также фирменное наименование Общества на любом языке народов Российской Федерации и (или) иностранном языке, а также присвоенный Обществу ИНН.

7. Общество вправе иметь штампы и бланки со своим фирменным наименованием, собственную эмблему, а также зарегистрированный в установленном виде товарный знак и другие средства индивидуализации и визуальной идентификации.

8. В соответствии с действующим законодательством Российской Федерации Общество вправе открывать расчетные и иные счета в кредитных организациях, как на территории Российской Федерации, так и на территории иностранных государств.

9. Общество вправе открывать филиалы, создавать представительства и иные обособленные подразделения, а также участвовать в капитале других юридических лиц, как на территории Российской Федерации, так и на территории иностранных государств, в соответствии с положениями настоящего Устава и действующим законодательством Российской Федерации с учетом особенностей, устанавливаемых нормами законодательства иностранного государства и нормами международного частного права.

Статья 4. Предмет деятельности.

1.Основной задачей Общества является организация рентабельной деятельности Общества и получение прибыли, используемой для развития Общества, расширения сферы его деятельности и укрепления финансового положения.

Общество вправе осуществлять любые виды деятельности, не запрещенные законодательством.

2. Предметом деятельности Общества являются:

- Разработка компьютерного программного обеспечения
- Деятельность консультативная и работы в области компьютерных технологий
- Деятельность, связанная с использованием вычислительной техники и информационных технологий, прочая
- Деятельность по обработке данных, предоставление услуг по размещению информации и связанная с этим деятельность
- Деятельность по созданию и использованию баз данных и информационных ресурсов
- Ремонт компьютеров и периферийного компьютерного оборудования;

и другие виды деятельности, не запрещенные действующим законодательством РФ.

Рисунок А.3 - Правовое положение и предмет деятельности

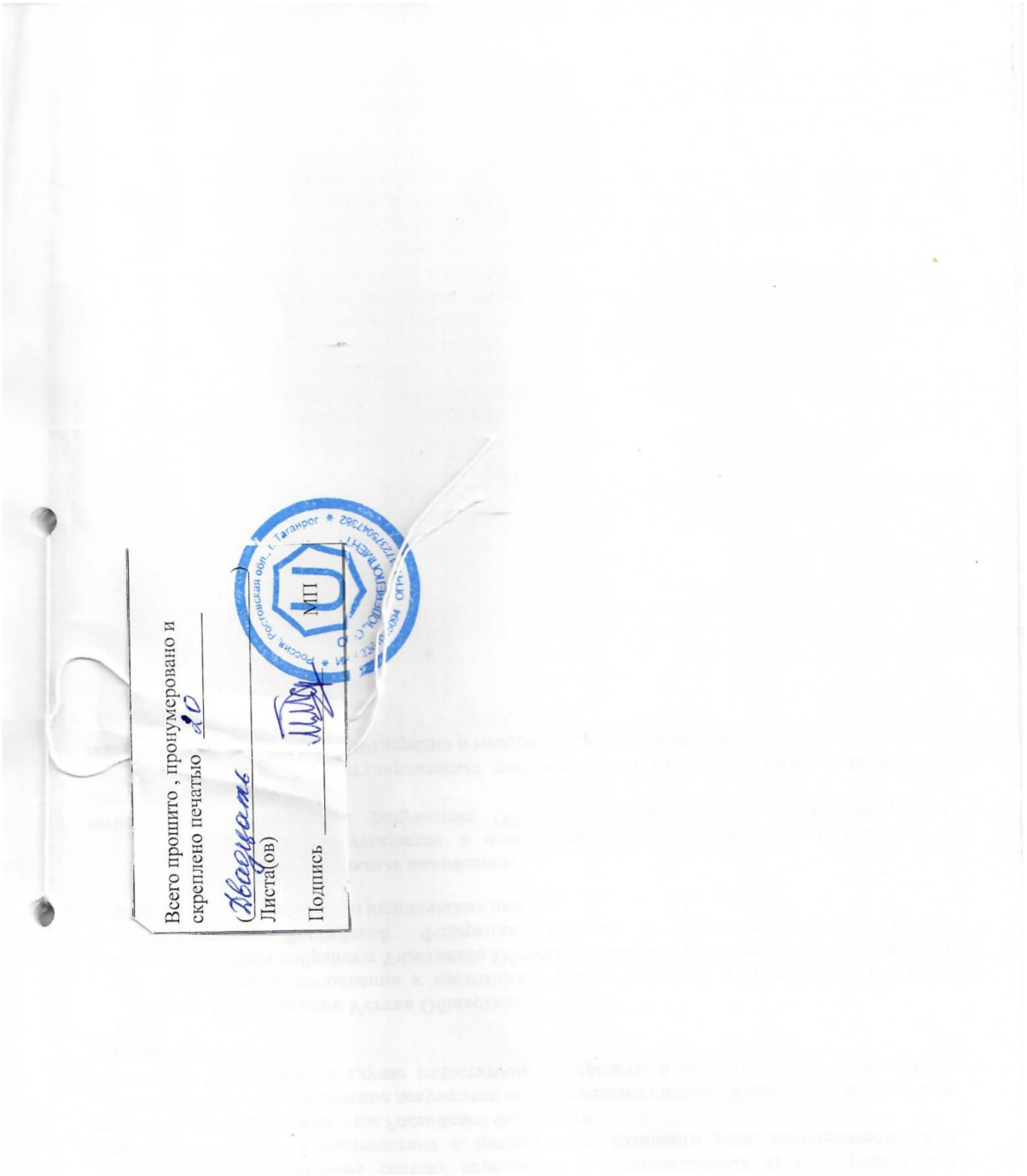


Рисунок А.4 - Печать предприятия на уставе

Приложение Б

(справочное)

Штатное расписание ООО «ЮДЕВЕЛОПМЕНТ»

(с. 63 — 64)

Унифицированная форма N Т-3
Утверждена постановлением Госкомстата
России от 05.01.2004 N 1

	Код
Форма по ОКУД	0301017
по ОКПО	00000000

«ЮДЕВЕЛОПМЕНТ»

ШТАТНОЕ РАСПИСАНИЕ

Номер документа	Дата составления
28	29.02.2017

УТВЕРЖДЕНО

на период 1 год с " 1 " марта 20 17 г.

Приказом организации от " 29 " февраля 20 17 года N 90

Штат в количестве 50 единиц

Структурное подразделение		Должность (специальность, профессия), разряд, класс (категория) квалификации	Коли-чество штатных единиц	Тарифная ставка (оклад) и пр., руб	Надбавки, руб			Всего, руб (гр.5 +гр.6+гр.7 +гр.8) х гр.4	Примечание
наименование	код								
1	2	3	4	5	6	7	8	9	10
Администрация	01	Директор	1	90000	0	0	0	90000	
Администрация	01	Заместитель директора	1	70000	0	0	0	70000	
Администрация	01	Бухгалтер	1	50000	0	0	0	50000	
Работа с персоналом	02	HR Менеджер	1	45000	0	0	0	45000	
Работа с персоналом	02	Офис менеджер	1	30000	0	0	0	30000	
Работа с клиентами	03	Менеджер проекта	4	40000	0	0	0	160000	
Работа с клиентами	03	Дизайнер	2	35000	0	0	0	70000	
Производственный тодел	04	Ведущий програмист	2	40000	0	0	0	80000	
Производственный тодел	04	Програмист	3	35000	0	0	0	105000	
Производственный тодел	04	Маладший програмист	1	30000	0	0	0	30000	
Производственный тодел	04	Ведущий мобильный програмист	4	40000	0	0	0	160000	
Производственный тодел	04	DevOps индженер	2	35000	0	0	0	70000	
Производственный тодел	04	Системный администратор	2	30000	0	0	0	60000	
Производственный тодел	04	Руководиель отдела DevOps	1	45000	0	0	0	45000	
Производственный тодел	04	Руководиель Web отдела	1	45000	0	0	0	45000	
Производственный тодел	04	Руководиель отдела мобильной разработки	1	45000	0	0	0	45000	
Отдел тестирования	05	Старший тестеровщик	2	30000	0	0	0	60000	
Отдел тестирования	05	Тестеровщик	3	25000	0	0	0	75000	
Отдел тестирования	05	Младший тестеровщик	5	20000	0	0	0	100000	
Финансовый отдел	06	Главный бухгалтер	1	30000	0	0	0	30000	
Финансовый отдел	06	Бухгалтер	1	25000	0	0	0	25000	
Администрация	01	Директор отдела работы с персоналом	1	50000	0	0	0	50000	
Администрация	01	Директор отдела работы с клиентоми	1	50000	0	0	0	50000	
Администрация	01	Директор производственного отдела	1	50000	0	0	0	50000	
Администрация	01	Директор отдела тестирования	1	50000	0	0	0	50000	
Администрация	02	Директор финфнсового	1	50000	0	0	0	50000	
		Итого	50	400000	0	0	0	595000	

Руководитель кадровой службы Директор отдела работы с персоналом (должность) — личная подпись — Макеев Игорь Васильевич (расшифровка подписи)

Главный бухгалтер — личная подпись — Шувалова Марина Игоревна (расшифровка подписи)

Рисунок Б.1 - Штатное расписание ООО «ЮДЕВЕЛОПМЕНТ»

Приложение В

(справочное)

Листинг фрагментов программного кода

(с. 65 — 95)

Класс Program

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace GoogleSpeech
{
class Program
   {
staticvoid Main(string[] args)
       {

Command cmd = newCommand();

string s;
string secLevel;
string log, pass = "";
ConsoleKeyInfo key ;

 log:       Interface.Authorization();
            log = Console.ReadLine();
            Console.Write("Пароль: ");
do
            {
                key = Console.ReadKey(true);
if (key.Key == ConsoleKey.Enter) break;
// Backspace Should Not Work
if (key.Key != ConsoleKey.Backspace)
                {
                    pass += key.KeyChar;
                    Console.Write("*");
                }
else
                {
                    Console.Write("\b");
                }
            }
while (key.Key != ConsoleKey.Enter);



//pass = Console.ReadLine();
```

```
if (cmd.Authorization(log, pass))
            {
goto start1;
            }
else
            {
                Console.Clear();
Console.WriteLine("Неверныйлогинилипароль!");
System.Threading.Thread.Sleep(2500);
goto log;
}


       start1:Console.WriteLine("Начало работы программы \"Виртуальный помошник
управления компьютером\" ");
start: Interface.PreviewUI();
              s = Console.ReadLine();

switch (s)
            {
case"Открытие программ":
start2: Interface.ProgrammUI();
                    secLevel = Console.ReadLine();
if (secLevel == "Назад") goto start;
                    cmd.SetCommand(secLevel);
                    cmd.ConstractCommand(secLevel);
if (cmd.OpenProgramm()) goto start2;
else
                    {
                        Console.WriteLine("<Комманда не выполнена>");
Console.ReadKey();
gotostart2;
                    }
break;




case"Управление компьютером":
                    Interface.SystemUI();
secLevel = Console.ReadLine();
                    cmd.SetCommand(secLevel);
                    cmd.ConstractCommand(secLevel);
                    cmd.ControlPC();
```

```
if (secLevel == "Назад")
                    {
goto start;
                    }

break;




case"Дополнительныеопции":
                    Interface.MoreUI();
                    secLevel = Console.ReadLine();

if (secLevel == "Назад")
                    {
goto start;
                    }
break;




default:
                    Console.WriteLine("\n<Неверныйввод>");
                    System.Threading.Thread.Sleep(1000);
goto start1;

}

if (s == "Открытие программ")
                {
            strt:


                }if (s == "Управление компьютером")
                {
                    Interface.SystemUI();
secLevel = Console.ReadLine();
```

```
if (secLevel == "Назад")
                    {
goto start;
                    }


               }if (s == "")
               {



if (secLevel == "Назад")
                    {
goto start;
                    }
               }
else
               {
                    Console.WriteLine("Ошибкаввода");
                    Console.Clear();
goto start;
               }

               Console.ReadKey();
               Console.Clear();
goto start;
       }
   }
}
```

Класс Json

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Runtime.Serialization.Json;
using System.Text;
using System.Runtime.Serialization;

namespace GoogleSpeech
{
publicclass JSon
   {
```

```
        [DataContract]
publicclass RecognizedItem
        {
            [DataMember]
publicstring utterance;

            [DataMember]
publicfloat confidence;
        }

        [DataContract]
publicclass RecognitionResult
        {
            [DataMember]
publicstring status;

            [DataMember]
publicstring id;

            [DataMember]
publicRecognizedItem[] hypotheses;
        }

publicstaticRecognitionResult Parse(String toParse)
        {
DataContractJsonSerializer                                  ser                                  =
newDataContractJsonSerializer(typeof(RecognitionResult));

MemoryStream stream1 = newMemoryStream(ASCIIEncoding.UTF8.GetBytes(toParse));

RecognitionResult result= (RecognitionResult)ser.ReadObject(stream1);
return result;
        }
    }

}
```

Класс GoogleVoice

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Net;
using System.Text;
```

```
namespace GoogleSpeech
{
publicclass GoogleVoice
   {

publicstaticString GoogleSpeechRequest(String flacName, int sampleRate)
     {
WebRequest        request        =        WebRequest.Create("https://www.google.com/speech-
api/v2/recognize?output=json&lang=ru-RU&key=AIzaSyDueoo-DR803XUfrwzWS2xeJ3_opbXHVF8");
       request.Method = "POST";

byte[] byteArray = File.ReadAllBytes(flacName);

// Set the ContentType property of the WebRequest.
       request.ContentType = "audio/x-flac; rate=" + sampleRate; //"16000";
       request.ContentLength = byteArray.Length;

// Get the request stream.
Stream dataStream = request.GetRequestStream();
// Write the data to the request stream.
       dataStream.Write(byteArray, 0, byteArray.Length);

       dataStream.Close();

// Get the response.
WebResponse response = request.GetResponse();

       dataStream = response.GetResponseStream();
// Open the stream using a StreamReader for easy access.
StreamReader reader = newStreamReader(dataStream);
// Read the content.
string responseFromServer = reader.ReadToEnd();

// Clean up the streams.
       reader.Close();
       dataStream.Close();
       response.Close();

return responseFromServer;
     }

publicstaticString GoogleSpeechRequest(String wavName, String flacName)
     {
int sampleRate = SoundTools.Wav2Flac(wavName, flacName);
return GoogleSpeechRequest(flacName, sampleRate);
```

```
    }
  }
}
```

Класс SoundTools

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using CUETools.Codecs;
using CUETools.Codecs.FLAKE;

namespace GoogleSpeech
{
publicstaticclass SoundTools
  {
/// <summary>Конвертированиewav-файлавoflac</summary>
/// <returns>Частотадискретизации</returns>
publicstaticint Wav2Flac(String wavName, string flacName)
      {
int sampleRate = 0;

IAudioSource audioSource = newWAVReader(wavName, null);
AudioBuffer buff = newAudioBuffer(audioSource, 0x10000);

FlakeWriter flakewriter = newFlakeWriter(flacName, audioSource.PCM);
          sampleRate = audioSource.PCM.SampleRate;

FlakeWriter audioDest = flakewriter;
while (audioSource.Read(buff, -1) != 0)
          {
              audioDest.Write(buff);
          }
          audioDest.Close();

          audioDest.Close();

return sampleRate;
      }
  }
}
```

Класс MainForm

```
using System;
```

```
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Net;
using System.Text;
using System.Windows.Forms;
using CUETools.Codecs;
using CUETools.Codecs.FLAKE;
using Newtonsoft.Json;
using NAudio.Wave;
using NAudio;
using System.Threading;

namespace GoogleSpeech
{
publicpartialclass MainForm : Form
   {
WaveIn waveIn;
WaveFileWriter writer;
string outputFilename = "demo.wav";
publicbool isRecord = false;
Thread sendFile;

public MainForm()
       {
           InitializeComponent();

           txtPath.Text = Settings.Instance.wavName;
       }

privatedelegatevoid DoWorkDelegate(String filePath);

privatevoid SendFileRun(String filePath)
       {
// Disable the button
         DisableStartButtons();
// Create delegate and make async call
DoWorkDelegate worker = newDoWorkDelegate(SendFile);
         worker.BeginInvoke(filePath, newAsyncCallback(DoWorkComplete), worker);
       }

privatevoid EnableStartButtons()
       {
```

```
if (InvokeRequired)
          {
            Invoke(newMethodInvoker(EnableStartButtons));
          }
else
          {
            btnBrowse.Enabled = true;
            btnSearch.Enabled = true;
          }

        }

privatevoid DisableStartButtons()
        {
          btnBrowse.Enabled = false;
          btnSearch.Enabled = false;
        }


privatevoid DoWorkComplete(IAsyncResult workID)
        {
          EnableStartButtons();
DoWorkDelegate worker = workID.AsyncState asDoWorkDelegate;
          worker.EndInvoke(workID);
          ReportOnProgress(100, "Запроспроизведен");
        }


privatedelegatevoid ReportOnProgressDelegate(int progress, string msg);

privatevoid ReportOnProgress(int progress, string msg)
        {
if (InvokeRequired)
          {
            Invoke(newReportOnProgressDelegate(ReportOnProgress),
newobject[] { progress, msg });
return;
            ;
          }

this.Text = Settings.Instance.AppTitle + " —  " + msg;
        }

privatevoid SendFile(String filePath)
        {
```

```
try
         {

           ReportOnProgress(10, "Идетзапрос");
String        responseFromServer        =        GoogleVoice.GoogleSpeechRequest(filePath,
Settings.Instance.tmpName);
           AddLog(responseFromServer + System.Environment.NewLine);

// JSon Обработка
var jsons = responseFromServer.Split('\n');

foreach (var j in jsons)
           {

dynamic jsonObject = JsonConvert.DeserializeObject(j);
if (jsonObject == null || jsonObject.result.Count <= 0) continue;

string text = jsonObject.result[0].alternative[0].transcript;
               AddToList(text);
           }

           ReportOnProgress(100, "Запросуспешновыполнен");
         }
catch (Exception e)
         {
           ReportOnProgress(100, "Ошибказапроса: " + e.Message);
           AddLog(e.ToString());
         }

       }

privatedelegatevoid AddDelegate(String log);
privatevoid AddLog(string log)
       {
if (InvokeRequired)
           {
             Invoke(newAddDelegate(AddLog), newobject[] { log });
return;
           }

         txtLog.Text += log;
       }

privatevoid AddToList(String item)
     {
if (InvokeRequired)
```

```
        {
          Invoke(newAddDelegate(AddToList), newobject[] { item });
return;
        }
        listBox1.Items.Add(item);
      }


privatevoid button1_Click(object sender, EventArgs e)
        {
            SendFileRun(Settings.Instance.wavName);
        }

privatevoid btnBrowse_Click(object sender, EventArgs e)
        {
            openFileDialog1.Title = "Open Wave File";
            openFileDialog1.Filter = "Wave files (*.wav)|*.wav";
if (openFileDialog1.ShowDialog() == System.Windows.Forms.DialogResult.Cancel)
            {
return;
            }

string sFilePath;
            sFilePath = openFileDialog1.FileName;
if (sFilePath == "")
            {
return;
            }

            Settings.Instance.wavName = sFilePath;
            txtPath.Text = sFilePath;
            SendFileRun(Settings.Instance.wavName);
        }

privatevoid textBox1_TextChanged(object sender, EventArgs e)
        {
            txtLog.SelectionStart = txtLog.Text.Length;
            txtLog.ScrollToCaret();
            txtLog.Refresh();
        }

privatevoid txtPath_TextChanged(object sender, EventArgs e)
        {
            Settings.Instance.wavName = txtPath.Text;
        }
```

```
privatevoid recBtn_Click(object sender, EventArgs e)
        {
if (isRecord)
            {
                waveIn.StopRecording();
                recBtn.Text = "Запись";
                isRecord = false;
            }
else {
//try
//{
                 waveIn = newWaveIn();
                waveIn.DeviceNumber = 0;
                waveIn.DataAvailable += waveIn_DataAvailable;
                waveIn.RecordingStopped                                                    +=
newEventHandler<NAudio.Wave.StoppedEventArgs>(waveIn_RecordingStopped);
                waveIn.WaveFormat = newWaveFormat(16000, 1);
                writer = newWaveFileWriter(outputFilename, waveIn.WaveFormat);
                waveIn.StartRecording();
                recBtn.Text = "Стоп";
                isRecord = true;
//}
//catch (Exception)
//{
//    MessageBox.Show("Упс, произошлаошибка");
//}

            }
        }

void waveIn_DataAvailable(object sender, WaveInEventArgs e)
        {
            writer.WriteData(e.Buffer, 0, e.BytesRecorded);
        }


void waveIn_RecordingStopped(object sender, EventArgs e)
        {
            waveIn.Dispose();
            waveIn = null;
            writer.Close();
            writer = null;
        }
    }
}
```

Класс Codecs

```
using System;
using System.IO;
using System.Text;
using System.Collections.Generic;
using System.Threading;
using System.Diagnostics;

namespace CUETools.Codecs
{
publicinterface IAudioSource
 {
int Read(AudioBuffer buffer, int maxLength);
void Close();

AudioPCMConfig PCM { get; }
string Path { get; }

long Length { get; }
long Position { get; set; }
long Remaining { get; }
 }

publicinterface IAudioDest
 {
void Write(AudioBuffer buffer);
void Close();
void Delete();

AudioPCMConfig PCM { get; }
string Path { get; }

int CompressionLevel { get; set; }
object Settings { get;  set; }
long FinalSampleCount { set; }
long BlockSize { set; }
long Padding { set; }
 }

publicinterface IAudioFilter
 {
IAudioDest AudioDest { set; }
 }
```

```
/// <summary>
///    This class provides an attribute for marking
///    classes that provide <seecref="IAudioDest"/>.
/// </summary>
/// <remarks>
///    When plugins with classes that provide <seecref="IAudioDest"/> are
///    registered, their <seecref="AudioEncoderClass"/> attributes are read.
/// </remarks>
/// <example>
///    <codelang="C#">using CUETools.Codecs;
///
///[AudioEncoderClass("libFLAC", "flac", true, "0 1 2 3 4 5 6 7 8", "5", 1)]
///public class MyEncoder : IAudioDest {
/// ...
///}</code>
/// </example>
 [AttributeUsage(AttributeTargets.Class, AllowMultiple = true)]
publicsealedclass AudioEncoderClass : Attribute
 {
privatestring _encoderName, _extension, _supportedModes, _defaultMode;
bool _lossless;
int _priority;
Type _settings;

public AudioEncoderClass(string encoderName, string extension, bool lossless, string
supportedModes, string defaultMode, int priority, Type settings)
   {
     _encoderName = encoderName;
     _extension = extension;
     _supportedModes = supportedModes;
     _defaultMode = defaultMode;
     _lossless = lossless;
     _priority = priority;
     _settings = settings;
   }

publicstring EncoderName
   {
get { return _encoderName; }
   }

publicstring Extension
   {
get { return _extension; }
   }
```

```
publicstring SupportedModes
    {
get { return _supportedModes; }
    }

publicstring DefaultMode
    {
get { return _defaultMode; }
    }

publicbool Lossless
    {
get { return _lossless; }
    }

publicint Priority
    {
get { return _priority; }
    }

publicType Settings
    {
get { return _settings; }
    }
 }

/// <summary>
///    This class provides an attribute for marking
///    classes that provide <seecref="IAudioSource"/>.
/// </summary>
/// <remarks>
///    When plugins with classes that provide <seecref="IAudioSource"/> are
///    registered, their <seecref="AudioDecoderClass"/> attributes are read.
/// </remarks>
/// <example>
///    <codelang="C#">using CUETools.Codecs;
///
///[AudioDecoderClass("libFLAC", "flac")]
///public class MyDecoder : IAudioSource {
/// ...
///}</code>
/// </example>
 [AttributeUsage(AttributeTargets.Class, AllowMultiple = false)]
publicsealedclass AudioDecoderClass : Attribute
 {
privatestring _decoderName, _extension;
```

```
public AudioDecoderClass(string decoderName, string extension)
   {
     _decoderName = decoderName;
     _extension = extension;
   }

publicstring DecoderName
   {
get { return _decoderName; }
   }

publicstring Extension
   {
get { return _extension; }
   }
 }

publicclass AudioPCMConfig
 {
privateint _bitsPerSample;
privateint _channelCount;
privateint _sampleRate;

public AudioPCMConfig(int bitsPerSample, int channelCount, int sampleRate)
   {
     _bitsPerSample = bitsPerSample;
     _channelCount = channelCount;
     _sampleRate = sampleRate;
   }

publicstaticreadonlyAudioPCMConfig RedBook = newAudioPCMConfig(16, 2, 44100);

publicint BitsPerSample { get { return _bitsPerSample; } }
publicint ChannelCount { get { return _channelCount; } }
publicint SampleRate { get { return _sampleRate; } }
publicint BlockAlign { get { return _channelCount * ((_bitsPerSample + 7) / 8); } }
publicbool IsRedBook { get { return _bitsPerSample == 16&& _channelCount == 2&&
_sampleRate == 44100; } }
 }

publicclass AudioBuffer
 {
privateint[,] samples;
privatefloat[,] fsamples;
privatebyte[] bytes;
```

```
privateint length;
privateint size;
privateAudioPCMConfig pcm;
privatebool dataInSamples = false;
privatebool dataInBytes = false;
privatebool dataInFloat = false;

publicint Length
   {
get
     {
return length;
     }
set
     {
       length = value;
     }
   }

publicint Size
   {
get
     {
return size;
     }
   }

publicAudioPCMConfig PCM { get { return pcm; } }

publicint ByteLength
   {
get
     {
return length * pcm.BlockAlign;
     }
   }

publicint[,] Samples
   {
get
     {
if (samples == null || samples.GetLength(0) < length)
         samples = newint[size, pcm.ChannelCount];
if (!dataInSamples && dataInBytes && length != 0)
         BytesToFLACSamples(bytes,    0,    samples,    0,    length,    pcm.ChannelCount,
pcm.BitsPerSample);
```

```
        dataInSamples = true;
return samples;
      }
    }

publicfloat[,] Float
    {
get
      {
if (fsamples == null || fsamples.GetLength(0) < length)
          fsamples = newfloat[size, pcm.ChannelCount];
if (!dataInFloat && dataInBytes && length != 0)
        {
if (pcm.BitsPerSample == 16)
            Bytes16ToFloat(bytes, 0, fsamples, 0, length, pcm.ChannelCount);
//else if (pcm.BitsPerSample > 16 && PCM.BitsPerSample <= 24)
//        BytesToFLACSamples_24(bytes,  0,  fsamples,  0,  length,  pcm.ChannelCount,  24  -
pcm.BitsPerSample);
elseif (pcm.BitsPerSample == 32)
            Buffer.BlockCopy(bytes, 0, fsamples, 0, length * 4 * pcm.ChannelCount);
else
thrownewException("Unsupported bitsPerSample value");
        }
        dataInFloat = true;
return fsamples;
      }
    }

publicbyte[] Bytes
    {
get
      {
if (bytes == null || bytes.Length < length * pcm.BlockAlign)
          bytes = newbyte[size * pcm.BlockAlign];
if (!dataInBytes && length != 0)
        {
if (dataInSamples)
            FLACSamplesToBytes(samples,    0,    bytes,    0,    length,    pcm.ChannelCount,
pcm.BitsPerSample);
elseif (dataInFloat)
            FloatToBytes(fsamples,     0,     bytes,     0,     length,     pcm.ChannelCount,
pcm.BitsPerSample);
        }
        dataInBytes = true;
return bytes;
      }
```

```
    }

public AudioBuffer(AudioPCMConfig _pcm, int _size)
    {
      pcm = _pcm;
      size = _size;
      length = 0;
    }

public AudioBuffer(AudioPCMConfig _pcm, int[,] _samples, int _length)
    {
      pcm = _pcm;
// assert _samples.GetLength(1) == pcm.ChannelCount
      Prepare(_samples, _length);
    }

public AudioBuffer(AudioPCMConfig _pcm, byte[] _bytes, int _length)
    {
      pcm = _pcm;
      Prepare(_bytes, _length);
    }

public AudioBuffer(IAudioSource source, int _size)
    {
      pcm = source.PCM;
      size = _size;
    }

publicvoid Prepare(IAudioDest dest)
    {
if    (dest.PCM.ChannelCount    !=    pcm.ChannelCount    ||    dest.PCM.BitsPerSample    !=
pcm.BitsPerSample)
thrownewException("AudioBuffer format mismatch");
    }

publicvoid Prepare(IAudioSource source, int maxLength)
    {
if   (source.PCM.ChannelCount   !=   pcm.ChannelCount   ||   source.PCM.BitsPerSample   !=
pcm.BitsPerSample)
thrownewException("AudioBuffer format mismatch");
      length = size;
if (maxLength >= 0)
        length = Math.Min(length, maxLength);
if (source.Remaining >= 0)
        length = (int)Math.Min((long)length, source.Remaining);
      dataInBytes = false;
```

```
      dataInSamples = false;
      dataInFloat = false;
    }

publicvoid Prepare(int maxLength)
    {
      length = size;
if (maxLength >= 0)
        length = Math.Min(length, maxLength);
      dataInBytes = false;
      dataInSamples = false;
      dataInFloat = false;
    }

publicvoid Prepare(int[,] _samples, int _length)
    {
      length = _length;
      size = _samples.GetLength(0);
      samples = _samples;
      dataInSamples = true;
      dataInBytes = false;
      dataInFloat = false;
if (length > size)
thrownewException("Invalid length");
    }

publicvoid Prepare(byte[] _bytes, int _length)
    {
      length = _length;
      size = _bytes.Length / PCM.BlockAlign;
      bytes = _bytes;
      dataInSamples = false;
      dataInBytes = true;
      dataInFloat = false;
if (length > size)
thrownewException("Invalid length");
    }

internalunsafevoid Load(int dstOffset, AudioBuffer src, int srcOffset, int copyLength)
    {
if (dataInBytes)
        Buffer.BlockCopy(src.Bytes, srcOffset * pcm.BlockAlign, Bytes, dstOffset *
pcm.BlockAlign, copyLength * pcm.BlockAlign);
if (dataInSamples)
        Buffer.BlockCopy(src.Samples, srcOffset * pcm.ChannelCount * 4, Samples, dstOffset
* pcm.ChannelCount * 4, copyLength * pcm.ChannelCount * 4);
```

```
if (dataInFloat)
       Buffer.BlockCopy(src.Float, srcOffset * pcm.ChannelCount * 4, Float, dstOffset *
pcm.ChannelCount * 4, copyLength * pcm.ChannelCount * 4);
   }

publicunsafevoid Prepare(AudioBuffer _src, int _offset, int _length)
   {
     length = Math.Min(size, _src.Length - _offset);
if (_length >= 0)
       length = Math.Min(length, _length);
     dataInBytes = false;
     dataInFloat = false;
     dataInSamples = false;
if (_src.dataInBytes)
       dataInBytes = true;
elseif (_src.dataInSamples)
       dataInSamples = true;
elseif (_src.dataInFloat)
       dataInFloat = true;
     Load(0, _src, _offset, length);
   }

publicvoid Swap(AudioBuffer buffer)
   {
if    (pcm.BitsPerSample    !=    buffer.PCM.BitsPerSample    ||    pcm.ChannelCount    !=
buffer.PCM.ChannelCount)
thrownewException("AudioBuffer format mismatch");

int[,] samplesTmp = samples;
float[,] floatsTmp = fsamples;
byte[] bytesTmp = bytes;

     fsamples = buffer.fsamples;
     samples = buffer.samples;
     bytes = buffer.bytes;
     length = buffer.length;
     size = buffer.size;
     dataInSamples = buffer.dataInSamples;
     dataInBytes = buffer.dataInBytes;
     dataInFloat = buffer.dataInFloat;

     buffer.samples = samplesTmp;
     buffer.bytes = bytesTmp;
     buffer.fsamples = floatsTmp;
     buffer.length = 0;
     buffer.dataInSamples = false;
```

```
      buffer.dataInBytes = false;
      buffer.dataInFloat = false;
    }

unsafepublicvoid Interlace(int pos, int* src1, int* src2, int n)
    {
if (PCM.ChannelCount != 2)
thrownewException("Must be stereo");
if (PCM.BitsPerSample == 16)
      {
        fixed (byte* bs = Bytes)
        {
          int* res = ((int*)bs) + pos;
for (int i = n; i >0; i--)
            *(res++) = (*(src1++) &0xffff) ^ (*(src2++) <<16);
        }
      }
elseif (PCM.BitsPerSample == 24)
      {
        fixed (byte* bs = Bytes)
        {
          byte* res= bs + pos * 6;
for (int i = n; i >0; i--)
          {
uint sample_out = (uint)*(src1++);
            *(res++) = (byte)(sample_out &0xFF);
            sample_out >>= 8;
            *(res++) = (byte)(sample_out &0xFF);
            sample_out >>= 8;
            *(res++) = (byte)(sample_out &0xFF);
            sample_out = (uint)*(src2++);
            *(res++) = (byte)(sample_out &0xFF);
            sample_out >>= 8;
            *(res++) = (byte)(sample_out &0xFF);
            sample_out >>= 8;
            *(res++) = (byte)(sample_out &0xFF);
          }
        }
      }
else
thrownewException("Unsupported BPS");
    }

//public void Clear()
//{
//    length = 0;
```

```
//}

publicstaticunsafevoid FLACSamplesToBytes_16(int[,] inSamples, int inSampleOffset,
     byte* outSamples, int sampleCount, int channelCount)
   {
int loopCount = sampleCount * channelCount;

if (inSamples.GetLength(0) - inSampleOffset < sampleCount)
thrownewIndexOutOfRangeException();

     fixed (int* pInSamplesFixed = &inSamples[inSampleOffset, 0])
     {
       int* pInSamples = pInSamplesFixed;
       short* pOutSamples = (short*)outSamples;
for (int i = 0; i < loopCount; i++)
         pOutSamples[i] = (short)pInSamples[i];
//*(pOutSamples++) = (short)*(pInSamples++);
     }
   }

publicstaticunsafevoid FLACSamplesToBytes_16(int[,] inSamples, int inSampleOffset,
byte[] outSamples, int outByteOffset, int sampleCount, int channelCount)
   {
int loopCount = sampleCount * channelCount;

if ((inSamples.GetLength(0) - inSampleOffset < sampleCount) ||
       (outSamples.Length - outByteOffset < loopCount * 2))
     {
thrownewIndexOutOfRangeException();
     }

     fixed (byte* pOutSamplesFixed = &outSamples[outByteOffset])
       FLACSamplesToBytes_16(inSamples,  inSampleOffset,  pOutSamplesFixed,  sampleCount,
channelCount);
   }

publicstaticunsafevoid FLACSamplesToBytes_24(int[,] inSamples, int inSampleOffset,
byte[] outSamples, int outByteOffset, int sampleCount, int channelCount, int wastedBits)
   {
int loopCount = sampleCount * channelCount;

if ((inSamples.GetLength(0) - inSampleOffset < sampleCount) ||
       (outSamples.Length - outByteOffset < loopCount * 3))
     {
thrownewIndexOutOfRangeException();
     }
```

```
      fixed (int* pInSamplesFixed = &inSamples[inSampleOffset, 0])
      {
        fixed (byte* pOutSamplesFixed = &outSamples[outByteOffset])
        {
          int* pInSamples = pInSamplesFixed;
          byte* pOutSamples = pOutSamplesFixed;

for (int i = 0; i < loopCount; i++)
          {
uint sample_out = (uint)*(pInSamples++) << wastedBits;
            *(pOutSamples++) = (byte)(sample_out &0xFF);
            sample_out >>= 8;
            *(pOutSamples++) = (byte)(sample_out &0xFF);
            sample_out >>= 8;
            *(pOutSamples++) = (byte)(sample_out &0xFF);
          }
        }
      }
    }

publicstaticunsafevoid FloatToBytes_16(float[,] inSamples, int inSampleOffset,
byte[] outSamples, int outByteOffset, int sampleCount, int channelCount)
    {
int loopCount = sampleCount * channelCount;

if ((inSamples.GetLength(0) - inSampleOffset < sampleCount) ||
        (outSamples.Length - outByteOffset < loopCount * 2))
      {
thrownewIndexOutOfRangeException();
      }

      fixed (float* pInSamplesFixed = &inSamples[inSampleOffset, 0])
      {
        fixed (byte* pOutSamplesFixed = &outSamples[outByteOffset])
        {
          float* pInSamples = pInSamplesFixed;
          short* pOutSamples = (short*)pOutSamplesFixed;

for (int i = 0; i < loopCount; i++)
          {
            *(pOutSamples++) = (short)(32758*(*(pInSamples++)));
          }
        }
      }
    }
```

```
publicstaticunsafevoid FloatToBytes(float[,] inSamples, int inSampleOffset,
byte[] outSamples, int outByteOffset, int sampleCount, int channelCount, int
bitsPerSample)
   {
if (bitsPerSample == 16)
       FloatToBytes_16(inSamples, inSampleOffset, outSamples, outByteOffset, sampleCount,
channelCount);
//else if (bitsPerSample > 16 && bitsPerSample <= 24)
//         FLACSamplesToBytes_24(inSamples, inSampleOffset, outSamples, outByteOffset,
sampleCount, channelCount, 24 - bitsPerSample);
elseif (bitsPerSample == 32)
       Buffer.BlockCopy(inSamples, inSampleOffset * 4 * channelCount, outSamples,
outByteOffset, sampleCount * 4 * channelCount);
else
thrownewException("Unsupported bitsPerSample value");
   }

publicstaticunsafevoid FLACSamplesToBytes(int[,] inSamples, int inSampleOffset,
byte[] outSamples, int outByteOffset, int sampleCount, int channelCount, int
bitsPerSample)
   {
if (bitsPerSample == 16)
       FLACSamplesToBytes_16(inSamples, inSampleOffset, outSamples, outByteOffset,
sampleCount, channelCount);
elseif (bitsPerSample >16&& bitsPerSample <= 24)
       FLACSamplesToBytes_24(inSamples, inSampleOffset, outSamples, outByteOffset,
sampleCount, channelCount, 24 - bitsPerSample);
else
thrownewException("Unsupported bitsPerSample value");
   }

publicstaticunsafevoid FLACSamplesToBytes(int[,] inSamples, int inSampleOffset,
     byte* outSamples, int sampleCount, int channelCount, int bitsPerSample)
   {
if (bitsPerSample == 16)
       FLACSamplesToBytes_16(inSamples, inSampleOffset, outSamples, sampleCount,
channelCount);
else
thrownewException("Unsupported bitsPerSample value");
   }

publicstaticunsafevoid Bytes16ToFloat(byte[] inSamples, int inByteOffset,
float[,] outSamples, int outSampleOffset, int sampleCount, int channelCount)
   {
int loopCount = sampleCount * channelCount;
```

```
if ((inSamples.Length - inByteOffset < loopCount * 2) ||
       (outSamples.GetLength(0) - outSampleOffset < sampleCount))
thrownewIndexOutOfRangeException();

     fixed (byte* pInSamplesFixed = &inSamples[inByteOffset])
     {
       fixed (float* pOutSamplesFixed = &outSamples[outSampleOffset, 0])
       {
         short* pInSamples = (short*)pInSamplesFixed;
         float* pOutSamples = pOutSamplesFixed;
for (int i = 0; i < loopCount; i++)
           *(pOutSamples++) = *(pInSamples++) / 32768.0f;
       }
     }
   }

publicstaticunsafevoid BytesToFLACSamples_16(byte[] inSamples, int inByteOffset,
int[,] outSamples, int outSampleOffset, int sampleCount, int channelCount)
   {
int loopCount = sampleCount * channelCount;

if ((inSamples.Length - inByteOffset < loopCount * 2) ||
       (outSamples.GetLength(0) - outSampleOffset < sampleCount))
     {
thrownewIndexOutOfRangeException();
     }

     fixed (byte* pInSamplesFixed = &inSamples[inByteOffset])
     {
       fixed (int* pOutSamplesFixed = &outSamples[outSampleOffset, 0])
       {
         short* pInSamples = (short*)pInSamplesFixed;
         int* pOutSamples = pOutSamplesFixed;

for (int i = 0; i < loopCount; i++)
         {
           *(pOutSamples++) = (int)*(pInSamples++);
         }
       }
     }
   }

publicstaticunsafevoid BytesToFLACSamples_24(byte[] inSamples, int inByteOffset,
int[,] outSamples, int outSampleOffset, int sampleCount, int channelCount, int
wastedBits)
```

```
   {
int loopCount = sampleCount * channelCount;

if ((inSamples.Length - inByteOffset < loopCount * 3) ||
       (outSamples.GetLength(0) - outSampleOffset < sampleCount))
thrownewIndexOutOfRangeException();

     fixed (byte* pInSamplesFixed = &inSamples[inByteOffset])
     {
       fixed (int* pOutSamplesFixed = &outSamples[outSampleOffset, 0])
       {
         byte* pInSamples = (byte*)pInSamplesFixed;
         int* pOutSamples = pOutSamplesFixed;
for (int i = 0; i < loopCount; i++)
         {
int sample = (int)*(pInSamples++);
           sample += (int)*(pInSamples++) <<8;
           sample += (int)*(pInSamples++) <<16;
           *(pOutSamples++) = (sample << 8) >> (8 + wastedBits);
         }
       }
     }
   }

   public static unsafe void BytesToFLACSamples(byte[] inSamples, int inByteOffset,
int[,]  outSamples,  int  outSampleOffset,  int  sampleCount,  int  channelCount,  int
bitsPerSample)
   {
     if (bitsPerSample == 16)
       BytesToFLACSamples_16(inSamples,   inByteOffset,   outSamples,   outSampleOffset,
sampleCount, channelCount);
     else if (bitsPerSample >16&& bitsPerSample <= 24)
       BytesToFLACSamples_24(inSamples,   inByteOffset,   outSamples,   outSampleOffset,
sampleCount, channelCount, 24 - bitsPerSample);
     else
thrownewException("Unsupported bitsPerSample value");
   }
 }

 public class AudioSamples
 {
   unsafe public static void Interlace(int* res, int* src1, int* src2, int n)
   {
     for (int i = n; i >0; i--)
     {
       *(res++) = *(src1++);
```

```
    *(res++) = *(src2++);
  }
}

unsafe public static void Deinterlace(int* dst1, int* dst2, int* src, int n)
{
  for (int i = n; i >0; i--)
  {
    *(dst1++) = *(src++);
    *(dst2++) = *(src++);
  }
}

unsafe public static bool MemCmp(int* res, int* smp, int n)
{
  for (int i = n; i >0; i--)
    if (*(res++) != *(smp++))
      return true;
  return false;
}

unsafe public static void MemCpy(uint* res, uint* smp, int n)
{
  for (int i = n; i >0; i--)
    *(res++) = *(smp++);
}

unsafe public static void MemCpy(int* res, int* smp, int n)
{
  for (int i = n; i >0; i--)
    *(res++) = *(smp++);
}

unsafe public static void MemCpy(long* res, long* smp, int n)
{
  for (int i = n; i >0; i--)
    *(res++) = *(smp++);
}

unsafe public static void MemCpy(short* res, short* smp, int n)
{
  for (int i = n; i >0; i--)
    *(res++) = *(smp++);
}

unsafe public static void MemCpy(byte* res, byte* smp, int n)
```

```
{
  if ((((IntPtr)smp).ToInt64() &7) == (((IntPtr)res).ToInt64() &7) && n >32)
  {
    int delta = (int)((8 - (((IntPtr)smp).ToInt64() &7)) &7);
    for (int i = delta; i >0; i--)
      *(res++) = *(smp++);
    n -= delta;

    MemCpy((long*)res, (long*)smp, n >>3);
    int n8 = (n >>3) <<3;
    n -= n8;
    smp += n8;
    res += n8;
  }
  if ((((IntPtr)smp).ToInt64() &3) == (((IntPtr)res).ToInt64() &3) && n >16)
  {
    int delta = (int)((4 - (((IntPtr)smp).ToInt64() &3)) &3);
    for (int i = delta; i >0; i--)
      *(res++) = *(smp++);
    n -= delta;

    MemCpy((int*)res, (int*)smp, n >>2);
    int n4 = (n >>2) <<2;
    n -= n4;
    smp += n4;
    res += n4;
  }
  for (int i = n; i >0; i--)
    *(res++) = *(smp++);
}

unsafe public static void MemSet(int* res, int smp, int n)
{
  for (int i = n; i >0; i--)
    *(res++) = smp;
}

unsafe public static void MemSet(long* res, long smp, int n)
{
  for (int i = n; i >0; i--)
    *(res++) = smp;
}

unsafe public static void MemSet(byte* res, byte smp, int n)
{
  if (IntPtr.Size == 8&& (((IntPtr)res).ToInt64() &7) == 0&& smp == 0&& n >8)
```

```
      {
        MemSet((long*)res, 0, n >>3);
        int n8 = (n >>3) <<3;
        n -= n8;
        res += n8;
      }
      if ((((IntPtr)res).ToInt64() &3) == 0&& smp == 0&& n >4)
      {
        MemSet((int*)res, 0, n >>2);
        int n4 = (n >>2) <<2;
        n -= n4;
        res += n4;
      }
      for (int i = n; i >0; i--)
        *(res++) = smp;
    }

    unsafe public static void MemSet(byte[] res, byte smp, int offs, int n)
    {
      fixed (byte* pres = &res[offs])
        MemSet(pres, smp, n);
    }

    unsafe public static void MemSet(int[] res, int smp, int offs, int n)
    {
      fixed (int* pres = &res[offs])
        MemSet(pres, smp, n);
    }

    unsafe public static void MemSet(long[] res, long smp, int offs, int n)
    {
      fixed (long* pres = &res[offs])
        MemSet(pres, smp, n);
    }

    public const uint UINT32_MAX = 0xffffffff;
}
```

Общество с ограниченной ответственностью "Юдевелопмент"
(ООО " Юдевелопмент ")

ИНН/КПП 2361015094/236101001 ОГРН 1172375047382
Юридический адрес: 353680, Краснодарский край, Ейский р-н, г.Ейск, ул. Грушовая, д. 10
Телефон: +7 (952) 5826062

Справка
о результатах внедрения проектных решений,
разработанных в выпускной квалификационной работе
студента Компьютерных технологий
и информационной безопасности (КТиИБ)
ФГБОУ ВО «РГЭУ(РИНХ)»,
группы ПИZS-341, Исаева Артем Игоревича
(научный руководитель от кафедры
Информационных систем и прикладной информатики (ИС и ПИ) - доцент,
к.э.н., Мирошниченко И.И.)

В процессе работы над выпускной квалификационной работой по теме: «Разработка системы голосового управления компьютером для людей с ограниченными возможностями в Ростовском филиале ООО «Юдевелопмент»» студент Исаев А.И. принимал непосредственное участие в разработке проекта и самого программного средства, реализующего задачу выпускной квалификационной работы.

Полученные результаты нашли отражение в проектных материалах на разработку поставленной задачи.

В настоящее время разработанное программное средство находится в опытной эксплуатации для внедрения в ООО «Юдевелопмент».

Менеджер по персоналу **Галай С.С.**

17.01.2020г.