**Министерство энергетики, промышленности и связи ставропольского края**

**Государственное бюджетное профессиональное образовательное учреждение «Ставропольский колледж связи имени Героя Советского   
Союза В.А. Петрова»**

Специальность: 09.02.07 Информационные системы и программирование

|  |  |
| --- | --- |
|  | **Допустить к защите** |
|  | приказом № \_\_от «\_\_» \_\_\_\_\_\_\_20\_\_г. |
|  | Заместитель директора по учебно-воспитательной работе |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.В. Анищенко |

МП

**ДИПЛОМНАЯ РАБОТА**

на тему:

Разработка ИС услуг для “Инфоком-С” на языке программирования C#

Дипломник: Колесиков Захар Георгиевич

(ФИО полностью)

Курс 4 группа ИП-184

Руководитель ВКР*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(подпись)* *(инициалы, фамилия)*

Ставрополь, 2022

**Министерство энергетики, промышленности и связи ставропольского края**

**Государственное бюджетное профессиональное образовательное учреждение «Ставропольский колледж связи имени Героя Советского Союза В.А. Петрова»**

|  |  |  |
| --- | --- | --- |
|  |  | **УТВЕРЖДАЮ** |
|  |  | Председатель ЦК ВТ  \_\_\_\_\_\_\_\_\_\_\_\_\_/И.В. Ерёмина  « \_\_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_г. |

**ЗАДАНИЕ**

**на выпускную квалификационную работу**

Студенту 4 курса, ИП-184 группы, специальности 09.02.07

\_\_\_\_\_\_\_\_\_\_\_\_ Колесникову Захару Георгиевичу \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*( фамилия, имя, отчество полностью)*

Тема работы Разработка информационной системы услуг для «Инфоком-С» на языке программирования C#

*полное наименование темы ВКР, согласно программе ГИА и приказа о закреплении тем ВКР*

Утверждена приказом по ГБПОУ СКС № \_\_от «\_\_» \_\_\_\_\_\_\_\_20\_\_\_г.

Исходные данные к ВКР: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Перечень подлежащих разработке в ВКР вопросов или краткое содержание ВКР:\_\_\_\_Разработать базу данных , информационную систему и графический интерфейс для вывода, ввода и редактирования информации о услугах

Перечень приложений (***при необходимости***): Приложение 1

Рассмотрено на заседании ЦК\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Протокол № \_\_ от «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_ г.

Срок сдачи законченной работы «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_ г.

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(*подпись руководителя) ( инициалы, фамилия)*

Задание принял к исполнению:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 20\_\_­­\_ г.

(*подпись обучающегося)*

**КАЛЕНДАРНЫЙ ПЛАН**

**Работы над дипломной работой студента**

**Колесникова Захара Георгиевича**

*фамилия, имя, отчество*

|  |  |  |  |
| --- | --- | --- | --- |
| **№**  **п/п** | **Наименование этапов выполнения выпускной квалификационной работы** | **Сроки** | **Примечание** |
|  | Утверждение тематики ВКР |  |  |
|  | Закрепление тем ВКР, назначение руководителя |  |  |
|  | Подбор и анализ исходной информации, подготовка и утверждение плана ВКР |  |  |
|  | Выполнение теоретической части ВКР |  |  |
|  | Выдача задания на ВКР и преддипломную практику |  |  |
|  | Сбор информации и систематизация материала во время прохождения преддипломной практики |  |  |
|  | Оформление практической части ВКР |  |  |
|  | Оформление приложений |  |  |
|  | Согласование содержания ВКР, устранение замечаний руководителя |  |  |
|  | Выполнение презентационной части |  |  |
|  | Проверка содержания и утверждение ВКР руководителем |  |  |
|  | Получение отзыва руководителя ВКР |  |  |
|  | Рецензирование ВКР |  |  |
|  | Предоставление ВКР в учебную часть, на получение допуска к защите |  |  |
|  | Предварительная защита |  |  |
|  | Защита ВКР |  |  |

Студент(ка) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*подпись инициалы, фамилия*

Руководитель работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*подпись инициалы, фамилия*

**СОДЕРЖАНИЕ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Введение………………………………………………………….. | | | 5 |
| 1 | Теоретические и методологические аспекты разработки специализированного программного обеспечения……………. | | | 9 |
|  | 1.1 | | Этапы разработки и внедрения программного обеспечения.………………………………………………… | 9 |
| 2 | Постановка задачи, анализ требований и разработка спецификаций …………………………………………………… | | | 29 |
|  | 2.1 | Описание участков и контуров управления предприятия | | 29 |
|  | 2.2 | Обзор программных средств, используемых для автоматизации контроля рабочего времени сотрудников . | | 38 |
| 3 | Проектирование программного продукта | | | 46 |
|  | 3.1 | | Функциональное проектирование программного продукта…………………………………………………….. | 46 |
|  | 3.2 | | Обектно-ориентированное проектирование программного продукта……………………………………. | 48 |
| 4 | Реализация программного продукта…………….. | | | 50 |
|  | Заключение…………………………………………………... | | | 52 |
|  | Список информационных источников ………… | | | 54 |
|  | Приложения ………………………………………………… | | | 56 |

**Введение**

На современном этапе развития, информация, которую обрабатывает человек, увеличивается ежедневно. Становится невозможным говорить о том, что деятельность предприятия является эффективной, если его деятельность полностью, или же отдельно взятого подразделения не автоматизировано, а в работе не применяются современные технологии и системы электронного управления.

В настоящее время развитие информационных технологий позволяет избавить человека от необходимости выполнения огромного количества простых и сложных расчетов, что позволяет повысить точность и продуктивность труда. Поэтому возникает потребность в написании программного продукта с использованием информационных систем, которые решили бы существующие проблемы производительности.

На сегодняшний день системы электронного управления дают учреждению возможность хранить большой объем информации, сокращая время на обработку данных, тем самым делая работу эффективной и простой.

В современных условиях важной областью стало информационное обеспечение, которое состоит в сборе и переработке информации, необходимой для принятия обоснованных управленческих решений. Передача информации о услугах и деятельности фирмы на высший уровень управления и взаимный обмен информацией между всеми взаимосвязанными подразделениями фирмы осуществляются на базе современной электронно-вычислительной техники и других технических средств связи.

Актуальность заключается в том, что разработка программного обеспечения значительно бы облегчила функционирование любого предприятия, повысив качество его работы.

В качестве предметной области для рассматриваемой системы задана область услуг предприятия.

Объект исследования - процессы получения, хранения и использование информации сотрудниками компании.

Предмет исследования - закономерности получения, хранения и использование информации сотрудниками компании.

Целью данной дипломной работы является разработка информационной системы услуг для “Инфоком-С”.

Практическая значимость выпускной квалификационной работы состоит в том, что использование средств автоматизации сокращает трудозатраты, за счет упрощения операции пополнения и представления информации о услугах компании.

1. **Теоретические и методологические аспекты разработки специализированного программного обеспечения.**

**1.1 Этапы разработки и внедрения программного обеспечения.**

**Разработка требований.**

Требование – это спецификация того, что должно быть реализовано. В них описывается поведение системы или её атрибуты и они могут служить средствами ограничения в процессе разработки системы.

Бизнес-требования – высокоуровневая бизнес – цель организации или заказчика.

Бизнес – правило – это политика, стандарт или правило, которое определяет или ограничивает некоторые стороны бизнес – процессов. Это не требование к программному обеспечению, это источник нескольких типов требований.

Ограничение – ограничение выбора вариантов, которые при проектировании и разработке продукта доступны разработчику.

Внешнее требование к интерфейсу – это описание взаимодействия между программным обеспечением и пользователем.

Характеристика – Одна или несколько логически связанных возможностей системы, которая представляет собой ценность для пользователя.

Функциональные требования – описание требуемого поведения системы в определённых условиях.

Нефункциональные требования – описание свойства или особенности, которыми должна обладать система. Может являтся и ограничением, которое должно соблюдаться.

Более детальное рассмотрение каждого уровня:

Бизнес – требования описывают, почему организации нужна такая система, то есть цели, которые она хочет достичь используя данную систему.

Пользовательские требования описывают цели и задачи, которые пользователь должен иметь возможность выполнять с помощью продукта. Они описывают то, что пользователь должен иметь возможность делать с системой.

Функциональные требования определяют, каково должно быть поведение продукта в разнообразных условиях. Такие требования описывают в форме традиционных утверждений.

Так сложилось, что бизнес – требования и функциональные требования ложатся в основу технического задания на разработку программного обеспечения.

Системные требования описывают требования к продукту, которые содержат многие компоненты или подсистемы.

Бизнес – правила – включают в себя корпоративные политики, правительственные постановления, отраслевые стандарты и вычислительные алгоритмы. Они располагаются вне любой системы программного обеспечения, но зачастую они накладывают ограничения на функции системы.

Область разработки технических условий делится на разработку требований и на управление требованиями.

Выявление и сбор требований.

Это этап включающий в себя все действия, связанные с выявлением требований, таких как совещания, анализ документов, создание прототипов и т.д. К ключевым действиям относятся:

Определение классов ожидаемых пользователей продукта.

Понимание задач и целей, а так же бизнес – целей, которым соответствуют данные задачи.

Изучение среды, в которой, в будущем, будет использоваться новый продукт.

Взаимодействие с отдельными людьми для полного понимания их потребностей и ожидания в сфере качества.

**Анализ.**

Одним из этапов разработки программного обеспечения является процедура проведения всестороннего анализа выдвинутых заказчиком требований к создаваемому ПО, чтобы определить основные цели и задачи продукта. В рамках этой стадии происходит максимально эффективное сотрудничество клиента и исполнителя, в ходе обсуждения деталей проекта помогающих более конкретно сформулировать требования к ПО. Результатом проведения анализа становится формирование технического задания, на которое будет опираться исполнитель при разработке ПО.

Как правило, этапу составления технического задания предшествует проведение обследования предметной области, которое завершается созданием аналитического отчета. Именно аналитический отчет (или аналитическая записка) ложится в основу документа Техническое задание.

Чем подробнее будет составлено техническое задание, тем меньше спорных ситуаций возникнет между заказчиком и разработчиком во время приемочных испытаний.

ГОСТ 19.201-78 «Техническое задание. Требования к содержанию и оформлению» является руководствующим стандартом при написании ТЗ.

Техническое задание должно содержать следующие разделы:

−введение;

−основания для разработки;

−назначение разработки;

−требования к программе или программному изделию;

−требования к программной документации;

−технико-экономические показатели;

−стадии и этапы разработки;

−порядок контроля и приемки

Главная цель технического задания – удостовериться, что заказчик и исполнитель правильно поняли друг друга.

Пользы от технического задания очень много. Для каждой стороны польза своя.

Польза для заказчика:

-**Понять, за что заплачены деньги**. Есть возможность сразу увидеть структуру, ознакомится с тем, что и как будет работать. Пересмотреть идеи ещё до начала разработки, чтобы не тратить лишний раз и своё, и чужое время.

**-Проверить компетентность исполнителя.**Если техническое задание чётко сформулировано, доверие к разработчику повышается. Если написанное мало похоже на техническое задание, возможно, стоит отказаться от услуг данного исполнителя.

**-Подстраховаться.**Когда сайт готов, его можно проверить по техническому заданию. Если имеются несоответствия, то исполнитель обязан их исправить.

**-Упростить замену исполнителей.**Если заказчик и исполнитель прекратили сотрудничать друг с другом, создание сайта может затянуться. Подробное техническое задание можно передать новой команде. Так она включится в работу в несколько раз быстрее.

**-Узнать стоимость разработки сложного продукта.**Оценить точные сроки и стоимость разработки сложного WEB–сервиса сразу не получится понять. Сначала нужно понять, как будет работать сервис, и какие в нем будут функции. Для этого и нужно подготовить техническое задание.

**Польза для исполнителя**

**-Понять, чего хочет заказчик.**Заказчику задают вопросы, относительно разработки, показывают примеры, предлагают решения. Затем записывают всё в единый документ и согласовывают.

**-Подстраховаться.** Иногда попадаются заказчики, которые хотят поменять задачу на полпути. Если вы согласовали и подписали техническое задание, подобное не произойдёт.

**-Показать свою компетентность.**Классно подготовленное техническое задание покажет клиенту умения, опыт и знания разработчиков. Если компания сомневалась, доверять ли вам разработку, то данные сомнения исчезнут.

**-Заработать.**Некоторые студии и разработчики предлагают составление ТЗ, как отдельную услугу.

-**Облегчить и ускорить разработку.**В хорошем техническом задании указаны структура, необходимые функции и элементы на каждой странице.

Хорошее техническое задание всегда составляет исполнитель – проект-менеджер или разработчик. Очевидно, что WEB-разработчик понимает в создании сайтов больше, чем владелец бизнеса, поэтому описывать проект придется ему.

Заказчик также должен участвовать в разработке технического задания:

- познакомить исполнителя с компанией, продуктами и целевой аудиторией;

- объяснить, зачем ему сайт;

- рассказать, чего хочет, поделиться идеями;

- показать примеры сайтов, которые можно взять за образец;

- ответить на любые вопросы исполнителя.

Заказчик тоже имеет полное право написать техническое задание, но с некоторой долей вероятности оно не пригодится.

**Проектирование.**

Следующим этапом разработки ПО является этап проектирования.

Этап проектирования представляет собой моделирование теоретической основы будущего продукта. Качественный анализ создаваемого продукта станет основой для полноценного функционирования и выполнения всех возложенных на ПО задач. Одной из частей этапа проектирования является выбор инструментальных средств и операционной системы.

Основным этапом проектирования является разработка диаграмм. Ramus является средством для сбора информации о компании и представления его в графическом виде. С помощью Ramus можно строить диаграммы благодаря которым можно провести анализ функциональности системы.

Контекстная диаграмма–это диаграмма, расположенная на вершине древовидной структуры диаграмм, представляющая собой самое общее описание системы и ее взаимодействие с внешней средой. Контекстная диаграмма состоит из одного блока, описывающего функцию верхнего уровня, ее входы, выходы, управления, и механизмы, вместе с формулировками цели модели и точки зрения, с которой строится модель.

Диаграммы декомпозиции предназначены для детализации функций и получаются при разбиении контекстной диаграммы на крупные подсистемы и описывающие каждый подсистему и их взаимодействие.

IDEF0-нотация графического моделирования, используемая для создания функциональной модели, отображающей структуру и функции системы, а также потоки информации и материальных объектов, связывающих эти функции.

Единственная функция, представленная на контекстной диаграмме верхнего уровня, может быть разложена на основные подфункции посредством создания дочерней диаграммы. В свою очередь, каждая из этих подфункций может быть разложена на составные части посредством создания дочерней диаграммы следующего, более низкого уровня, на которой некоторые или все функции также могут быть разложены на составные части. Каждая дочерняя диаграмма содержит дочерние блоки и стрелки, обеспечивающие дополнительную детализацию родительского блока.

**Программирование.**

Следующим этапом является программирование, опираясь на заранее выбранный язык программирования. В этом этапе нужно написать код, который реализует выполнение ранее описанного алгоритма. Кодирование может происходить параллельно следующему этапу разработки ПО. Следующем этапом разработки является тестирование и при параллельном кодировании можно вносить изменения в код по ходу работы. Этап кодирования очень сильно отражает качество предыдущих этапов разработки программного обеспечения. После кодирования проводится отладка, которая выявляет все ошибки присутствующие в коде, которые необходимо исправить для корректной работы программного обеспечения.

Системные администраторы настраивают программное окружение, программисты разрабатывают интерфейс программы и её логику совместной работы с сервером.

Программирование предполагает четыре основные стадии:

* Разработка алгоритмов
* Написание исходного кода
* Компиляция
* Тестирование и отладка

**Тестирование.**

Этап тестирования подразумевает под собой проверку всего функционала программного продукта. Во время тестирования, зачастую, проверяют каждый компонент по отдельности, т.к. очень крупные программы имеют огромный функционал, который делят на компоненты, разработка которых проводится отдельно. В менее крупных проектах данный этап может включать в себя тестирование отдельных классов.

Есть несколько широко используемых методов тестирования ПО, которые используются в определённом порядке:

1. Модульное тестирование;
2. Интеграционное тестирование;
3. Системное тестирование;
4. Приёмочные испытания;

Выделяют несколько принципов тестирования:

1 Демонстрация наличия дефектов.

- Тестирование показывает присутствие дефектов, но не может доказать, что их нет. Тестирование уменьшает шанс наличия дефектов, находящихся в программном обеспечении, но даже если дефекты не были обнаружены, то тестирование не доказывает корректность программного обеспечения.

2 Исчерпывающее тестирование недостижимо.

- Абсолютное тестирование с использованием всевозможных комбинаций вводов и предисловий не возможно, кроме тривиальных случаев.

За место попытки исчерпывающего тестирования должны использоваться анализ рисков, методов тестирования и расстановка приоритетов, что бы сосредоточить усилия на тестировании.

3 Ранее тестирование сохраняет время и деньги.

- Активность тестирования должна начинаться как можно раньше в жизненном цикле разработки программного продукта. Это даёт возможность находить дефекты на ранних стадиях. Тестирование на ранних этапах жизненного цикла программного обеспечения позволяет избежать дорогостоящие изменения, т.к. вовремя замеченную ошибку в техническом задании намного проще избежать или исправить, чем когда по этому техническому заданию будет разработан функционал.

4 Кластеризация дефектов.

- Обычно, большинство дефектов содержаться в небольшом количестве модулей, которые обнаруживаются во время тестирования перед выпуском программного обеспечения. Дефекты имеют свойство скапливаться в одном месте и, если нашли интересную ошибку в функционале, то есть большая вероятность найти ещё одну поблизости.

5 Эффект пестицида.

- Если часто использовать одни и те же способы тестирования, то со временем больше не будут находиться новые дефекты. Для нахождения новых ошибок можно быть востребованным изменение существующих тестов и тестовых данных, или же написание новых тестов.

6 Тестирование зависит от контекста.

- Тестирование выполняется в зависимости от контекста, например тестирование программного обеспечения управления производством, где очень важна безопасность, проходит иначе по сравнению с тестированием мобильного приложения.

7 Заблуждение об отсутствии ошибок.

- Ошибочно ждать успех продукта, если было найдена и исправлена большая часть дефектов, ведь при данной ситуации программное обеспечение может стать трудной в использовании и не будет соответствовать ожиданиям и потребностям пользователя.

**Внедрение.**

Внедрение ПО в эксплуатацию – это завершающая стадия разработки и не редко проходит совместно с отладкой системы. Как правило, ввод в эксплуатацию проводится в 3 этапа:

1. Первоначальная загрузка
2. Постепенное накопление информации
3. Ввод созданного ПО на проектную мощность

Ключевой целью поэтапного введения программного обеспечения является постепенное выявление ранее не обнаруженных ошибок и недочётов кода. Именно на этой стадии вырисовывается окончательная картина взаимодействия пользователя с программой, а также определяется степень лояльности пользователя к разработанному интерфейсу.

Когда выходит новая версия приложения к работе присоединяется отдел технической поддержки. Данный отдел обеспечивает связь с пользователями, их консультирование и поддержку.

В случае обнаружения пользователями тех или иных проблем в работе программы, информация о них передаётся в виде отчётов команде разработчиков, которая, в зависимости от серьёзности проблемы, решает выпускать исправление не медленно, или же откладывает до следующей версии программы.

Так же, техническая поддержка, помогает собирать и систематизировать различные показатели работы программы в реальных условиях.

**2 Постановка задачи, анализ требований и разработка спецификаций**

**2.1 Описание участков и контуров управления предприятия**

Компания ООО «Инфоком-С» была основана в 2012 году как малое инновационное предприятие при крупном вузе. На сегодняшний день ООО «Инфоком-С» представляет собой развивающуюся компанию, специализирующуюся на научных исследованиях и опытно-конструкторских разработках, а также наукоемких проектах по созданию и внедрению сложных программно-технических систем.

Специализацией компании являются инновационные ИТ – продукты. Компания готова предложить своим клиентам инновационные продукты и обеспечить качественное внедрение в существующую ИТ – инфраструктуру.

Квалификация и опыт работы сотрудников «Инфоком-С» имеет очень высокие для региона опыт и квалификацию для развития информационного общества.

Важными направлениями в работе являются аналитические и научные исследования. Компания работает с фундаментальными и прикладными научными исследованиями в сфере информационных технологий, реализуя полученные знания и опыт в своих проектах.

Огромный опыт компании, связанный с большим количеством проектной деятельности, содержит в себе теоретические и прикладные исследования, проводимые научными сотрудниками в сфере информационных технологий.

Профессиональные опыт и знания дают возможность выполнять комплекс опытно-конструкторских работ – начиная от анализа предметной области с формированием требований к предмету разработки и заканчивая подготовкой к передачи в производство.

Данная команда высококвалифицированных сотрудников обладает бесценным опытом. Они создают новые научно-технические решения, благодаря чему компания успешно конкурирует на рынке информационных технологий и достигает высоких финансово-экономических показателей. За годы работы они реализовали потенциал некоторых ведущих учёных и юных, креативно мыслящих специалистов.

**2.2 Разработка технического задания**

Техническое задание это документация, содержащая требования заказчика к объекту закупки, определяющие условия и порядок ее проведения, в соответствии с которыми осуществляются поставка товара, выполнение работ, оказание услуг и их приём. Техническое задание используется в качестве исходной документации, в которой описываются такие аспекты как - основное назначение закупки товаров, работ, услуг, их характеристики, задание заказчика, описание первичных данных, целей и задач закупки, сроков поставки, выполнения работ, оказания услуг, требований к товару, работам, услугам, их результатам, к гарантиям, описание объекта закупки, объем закупаемых товаров, работ, услуг, формы отчетности, обоснование требований к товару, работам, услугам, эквивалентные показатели, экономические требования, а также специальные требования.

Техническое задание на разработку программного продукта приведено в приложении А.

**3 Проектирование программного продукта**

**3.1 Функциональное проектирование программного продукта**

Функциональными диаграммами, называют диаграммы , которые в первую очередь отражают взаимосвязи функций разрабатываемого программного обеспечения. В качестве примера рассмотрим модель, предложенную Д. Россом в составе методологии функционального моделирования SADT (Structured Analysis and Design Technique).

Методология SADT представляет собой совокупность методов, правил и процедур, предназначенных для построения функциональной модели объекта какой-либо предметной области. Функциональная модель SADT отображает функциональную структуру объекта, т.е. производимые им действия и связи между этими действиями. Основные элементы этой методологии основываются на следующих концепциях:

* графическое представление блочного моделирования. Графика блоков и дуг SADT-диаграммы отображает функцию в виде блока, а интерфейсы входа/выхода представляются дугами, соответственно входящими в блок и выходящими из него. Взаимодействие блоков друг с другом описываются посредством интерфейсных дуг, выражающих "ограничения", которые в свою очередь определяют, когда и каким образом функции выполняются и управляются;
* строгость и точность. Выполнение правил SADT требует достаточной строгости и точности, не накладывая в то же время чрезмерных ограничений на действия аналитика. Правила SADT включают:
* ограничение количества блоков на каждом уровне декомпозиции (правило 3-6 блоков);
* связность диаграмм (номера блоков);
* уникальность меток и наименований (отсутствие повторяющихся имен);
* синтаксические правила для графики (блоков и дуг);
* разделение входов и управлений (правило определения роли данных).
* отделение организации от функции, т.е. исключение влияния организационной структуры на функциональную модель.

На основе SADT разработана методология IDEF0 (Icam DEFinition).

Отображение взаимосвязи функций функциональной модели происходит с помощью выстраивания иерархии функциональных диаграмм, схематически представляющих взаимосвязи нескольких функций. К каждому блоку относится функция, для которой должны быть определены: исходные данные, результаты, управляющая информация и механизмы ее осуществления–человек или технические средства.

Основным этапом при проектировании информационной системы является разработка диаграмм. Для этого можно использовать Ramus, т.к. он является средством для сбора и графического изображения информации о предприятии. Благодаря Ramus можно построить диаграммы DFD и IDEF0 типа, с помощью которых проводится анализ функциональности систем.

Пять типов влияний блоков друг на друга в функциональных диаграммах SADT:

−вход-выход блока подается на вход блока с меньшим доминированием, т. е. следующего;

−управление-выход блока используется как управление для блока с меньшим доминированием (следующего);

−обратная связь по входу - выход блока подается на вход блока с большим доминированием (предыдущего);

−обратная связь по управлению—выход блока используется как управляющая информация для блока с большим доминированием (предыдущего);

−выход-исполнитель-выход блока используется как механизм для другого блока.

Контекстная диаграмма является одним видом IDEF0-диаграммы. Это диаграмма, расположенная на вершине древовидной структуры диаграмм, представляющая собой самое общее описание системы и ее взаимодействие с внешней средой. Контекстная диаграмма состоит из одного блока, описывающего функцию верхнего уровня, ее входы, выходы, управления, и механизмы, вместе с формулировками цели модели и точки зрения, с которой строится модель.

Декомпозиция блока контекстной диаграммы – это процесс создания детальных диаграмм - каждая функция, представленная отдельным блоком, может быть описана более подробно на другой диаграмме, расположенной на один уровень ниже в иерархии. Диаграмма нижнего уровня, или диаграмма-потомок, показывает внутреннее содержание блока-родителя.

Диаграммы декомпозиции предназначены для детализации функций и получаются при разбиении контекстной диаграммы на крупные подсистемы и описывающие каждый подсистему и их взаимодействие.

На рисунке 1 представлена контекстная диаграмма, представляющая систему целиком.

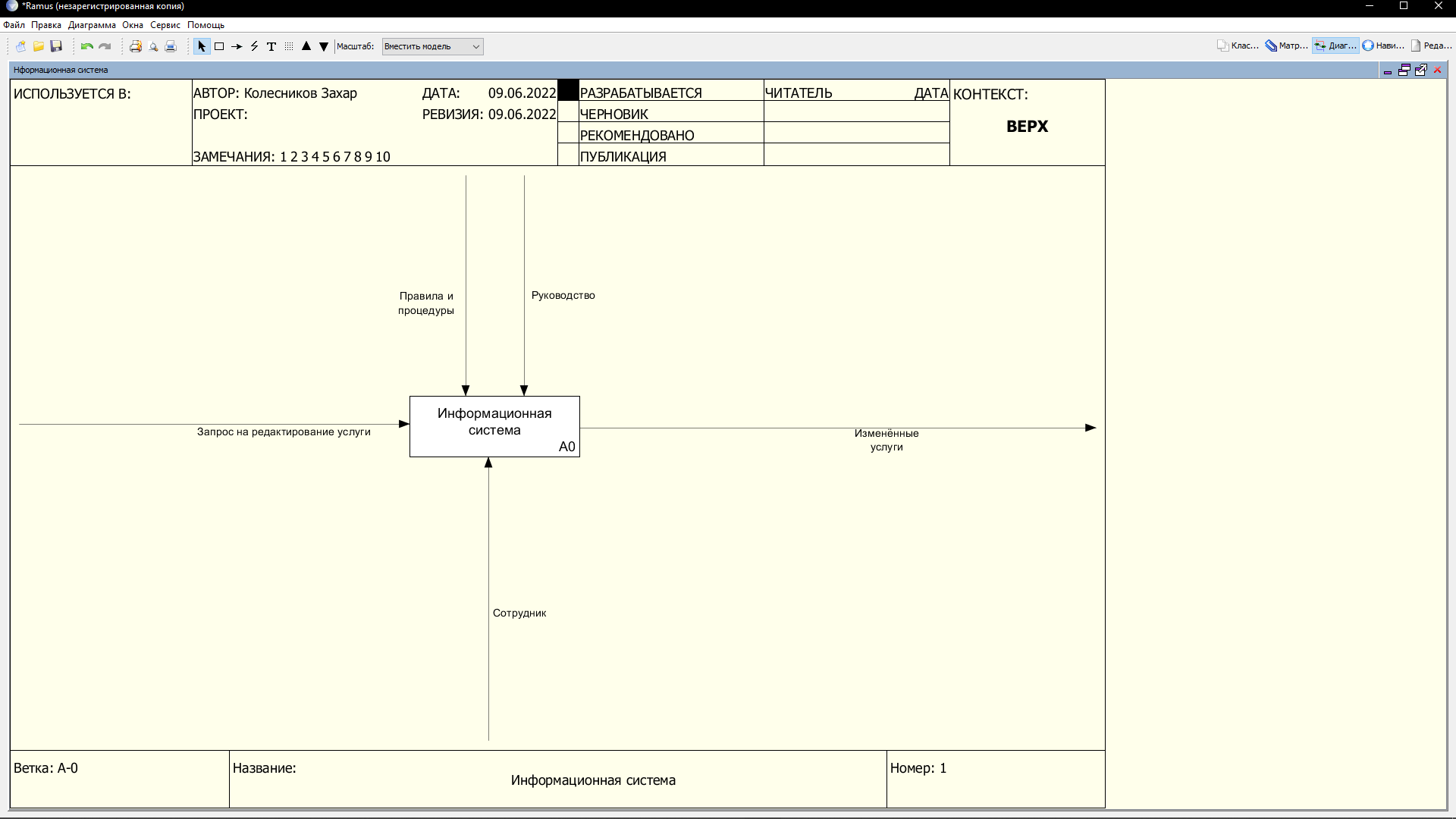


Рисунок 1 – Контекстная диаграмма проекта

На рисунке 2 представлена диаграмма декомпозиции в нотации IDEF0. На диаграмме представлено 4 функциональных блока, описывающих процесс работы разрабатываемого приложения.

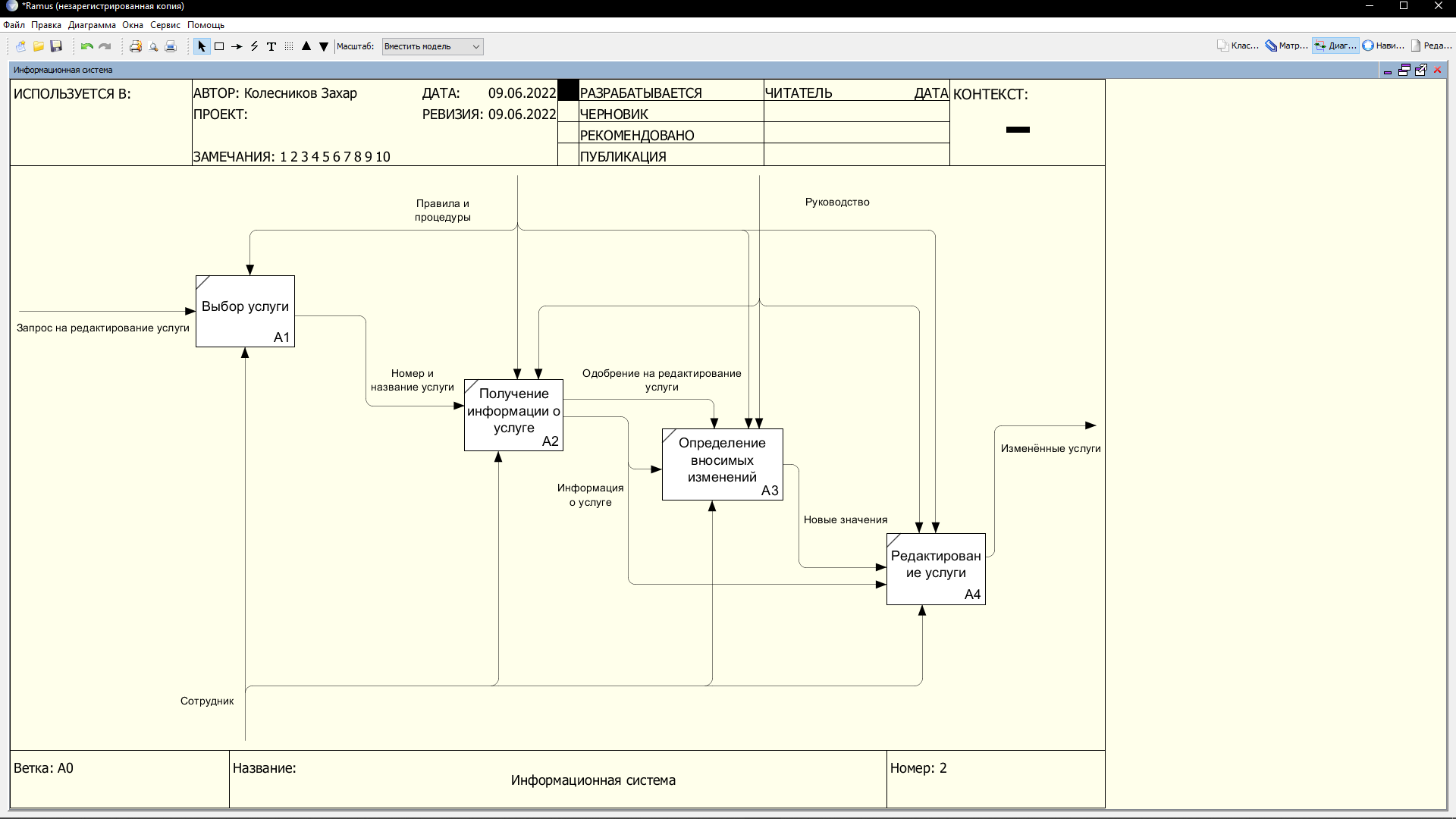


Рисунок 2 – Диаграмма декомпозиции в нотации IDEF0

* 1. **Объектно-ориентированное проектирование программного продукта**

Функциональные (структурные) технологии моделирования информационных систем характеризуются искусственным разделением системы на подсистемы, а также слабой взаимосвязью процессов и данных. В отличие от них, объектные технологии, ориентированные на тесную взаимосвязь процессов и данных, позволяют создавать более надежные, легкореализуемые и устойчивыми к изменениям программные системы. Кроме того, объектная философия моделирования в большей степени соответствует общим концепциям поведения систем реального мира. При модификации системы объектный подход позволяет легко включать в систему новые объекты и исключать устаревшие без существенного изменения ее структуры и жизнеспособности. Использование построенной модели при модификациях системы дает возможность устранить нежелательные последствия изменений, поскольку они не ломают устоявшейся структуры системы, а только немного изменяют её в нужном направлении.

Объектно-ориентированное проектирование имеет базис из пяти фундаментальных понятий:

* Объект;
* Класс;
* Инкапсуляция;
* Наследование;
* Полиморфизм;

Объектом в ООП называют программную модель некоторого реального или идеального объекта внешнего мира. Описание такого объекта в общем случае может включать ряд параметров, моделирующих состояние прототипного объекта и набор функций, предназначенных для обработки этих атрибутов с целью их изменения. Такая структура программного объекта даёт возможность моделировать процессы функционирования прототипных объектов, т.е. процессы, изменяющие их состояние во времени. ООП, таковым способом, позволяет моделировать динамические объекты и системы.

Классом называют набор объектов с одинаковыми атрибутами и методами, а объекты-члены этого набора–экземплярами класса. Класс, таковым способом, определяет тип своих объектов. В соответствии с принципами строгой типизации, в языках объектно-ориентированного программирования определение класса отделяется от определения его объектов.

Инкапсуляция–это объединение описаний атрибутов и методов всех однотипных объектов внутри описания соответствующего класса. При этом принимаются специальные меры по защите экземпляров класса от непреднамеренного изменения или разрушения в моделирующей программе. Инкапсуляция, таковым способом, позволяет:

* Быстро сократить размер программы за счёт однократного описания определяемых в программе объектов внутри описания соответствующего класса;
* Описания классов зачастую убирается из тела программы и могут храниться в отдельных файлах на дисковых накопителях, потому что, они используются только на этапе компилирования программы;
* Обеспечит сохранность объектов от непреднамеренного редактирования и удаления;

Наследование – это особый вид отношений иерархического подчинения между объектами «родитель - потомок» .  Основным свойством этого отношения является наследование потомком характеристик и поведения своего родителя. Реализация этого типа отношений является обязательным требованием при разработке языков объектно-ориентированного программирования. В контексте этих языков принцип наследования формулируется следующим образом: объект-потомок наследует все атрибуты и методы объектов родительского класса. Это означает, что в определении класса потомков атрибуты и методы родительского класса повторять не нужно–они действуют «по умолчанию». Потомок, однако, никогда не является точной и полной копией своих родителей: у него всегда проявляются какие-то индивидуальные свойства и поведение. Поэтому, в определения классов потомков, зачастую, добавляются свои собственные методы и атрибуты, которые отличаются от родительских. По мимо всего прочего, состояние потомка можно изменять не только вызовом его собственных методов, но и вызовом родительских методов.

Полиморфизм подразумевает под собой, что различные типы объектов, объединенные общим происхождением, могут иметь методы с одинаковыми названиями, но разной реализацией, так что при их активизации поведение разных объектов будет различным.

* + 1. **Диаграмма последовательности**

Диаграмма последовательности—диаграмма, на которой для некоторого набора объектов на единой временной оси показан жизненный цикл какого-либо определённого объекта и взаимодействие актеров ИС в рамках какого-либо определённого прецедента.

Основными элементами диаграммы последовательности являются обозначения объектов ,вертикальные «линии жизни», отображающие течение времени, прямоугольники, отражающие деятельность объекта или исполнение им определенной функции, и стрелки, показывающие обмен сигналами или сообщениями между объектами. На рисунке 3 Изображена диаграмма последовательности.

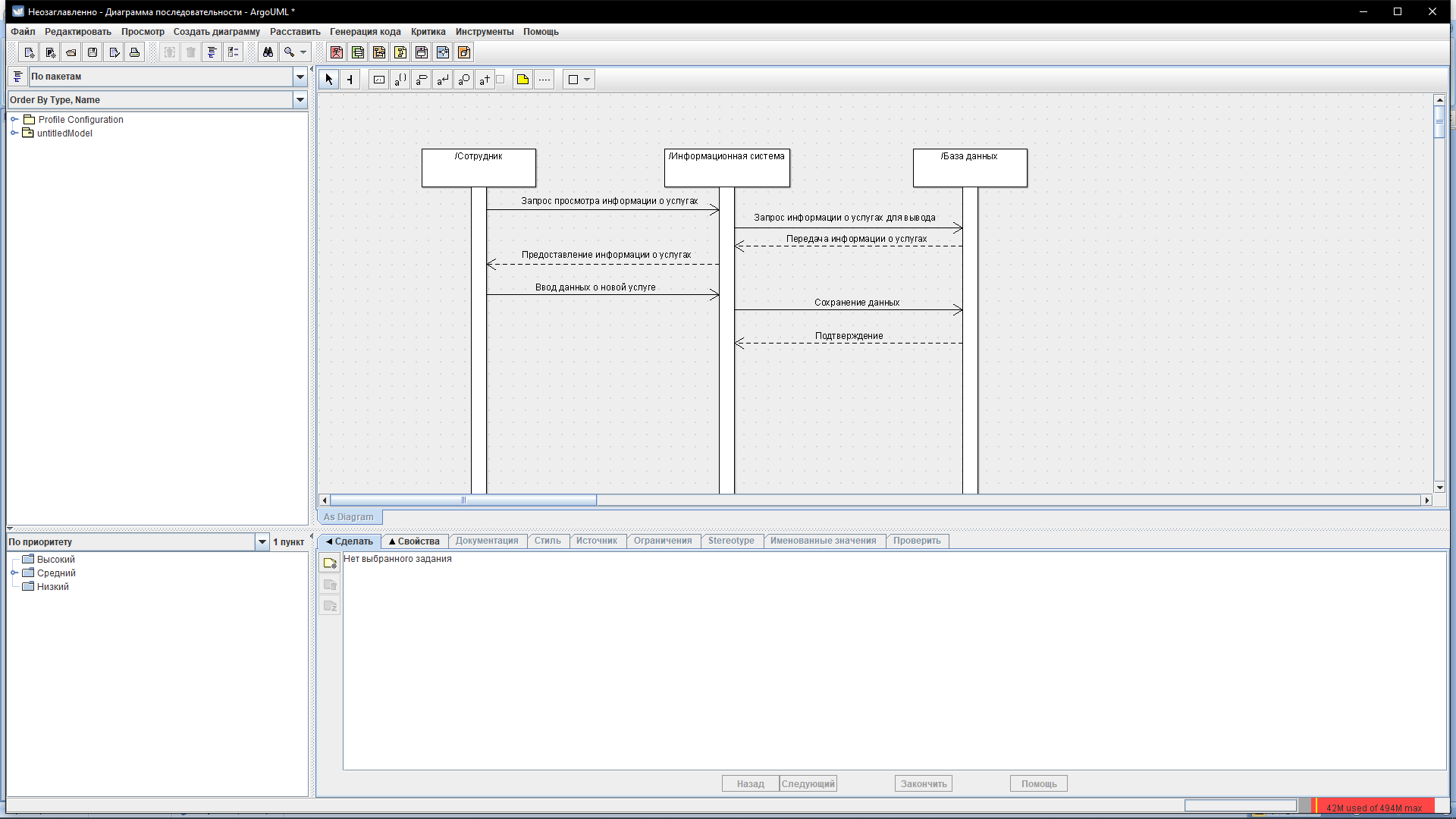
****

Рисунок 3 – Диаграмма последовательности

* + 1. **Диаграмма деятельности**

Диаграмма деятельности—диаграмма, на которой показаны действия, состояния которых описано на диаграмме состояний. Под деятельностью понимается спецификация исполняемого поведения в виде координированного последовательного и параллельного выполнения подчинённых элементов—вложенных видов деятельности и отдельных действий, соединённых между собой потоками, которые идут от выходов одного узла ко входам другого.

Диаграммы деятельности, изображённая на рисунке 4, используются при моделировании бизнес-процессов, технологических процессов, последовательных и параллельных вычислений.

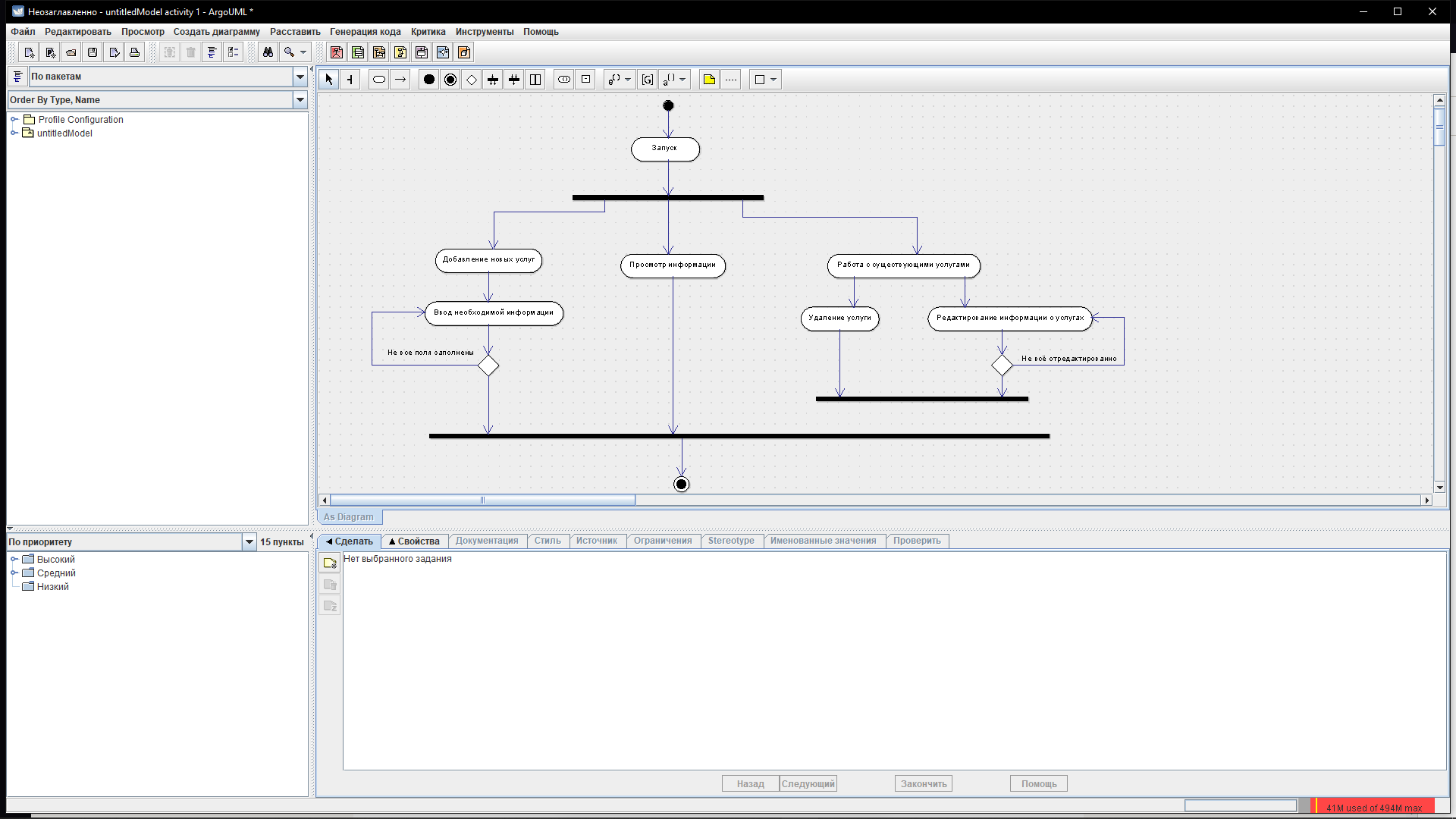
****

Рисунок 4 – Диаграмма последовательности

**4 Реализация программного продукта**

* 1. **Реализация интерфейса и программного кода приложения**

Один из главных этапов разработки приложения является реализация интерфейса приложения понятного будущему пользователю приложения.

Пользовательский интерфейс – это набор программных и аппаратных средств, обеспечивающих взаимодействие пользователя с компьютером. Основу такого взаимодействия составляет регламентированный обмен информацией между человеком и компьютером, осуществляемый в реальном времени и направленный на совместное решение поставленной задачи.

Реализованный в данной работе интерфейс применяется для добавления и редактирования услуг пользователем. Интерфейс главной формы представлен на рисунке 5. На данной форме имеются кнопки перехода на формы:

* Добавления новых услуг;
* Редактирование уже существующих услуг;
* Вывод формы справки;

Так же на данной форме реализовано окно просмотра базы данных, в которой и содержится вся информация о услугах.

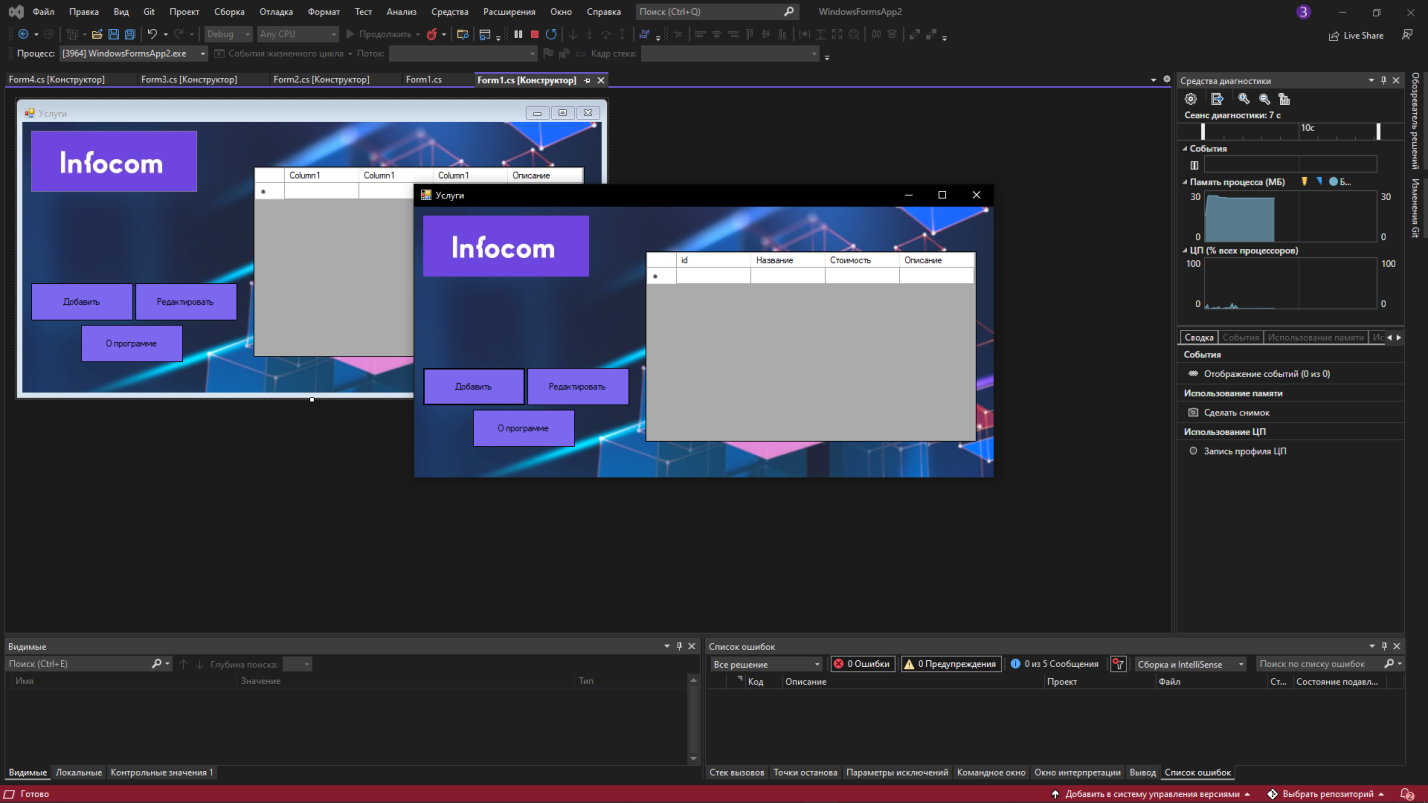


Рисунок 5 – Главная форма приложения

Форма добавления услуги представлена на рисунке 6. На данной форме располагаются поля для ввода информации. Кнопка «Добавить», как ни странно, добавляет все введённые данные в БД.

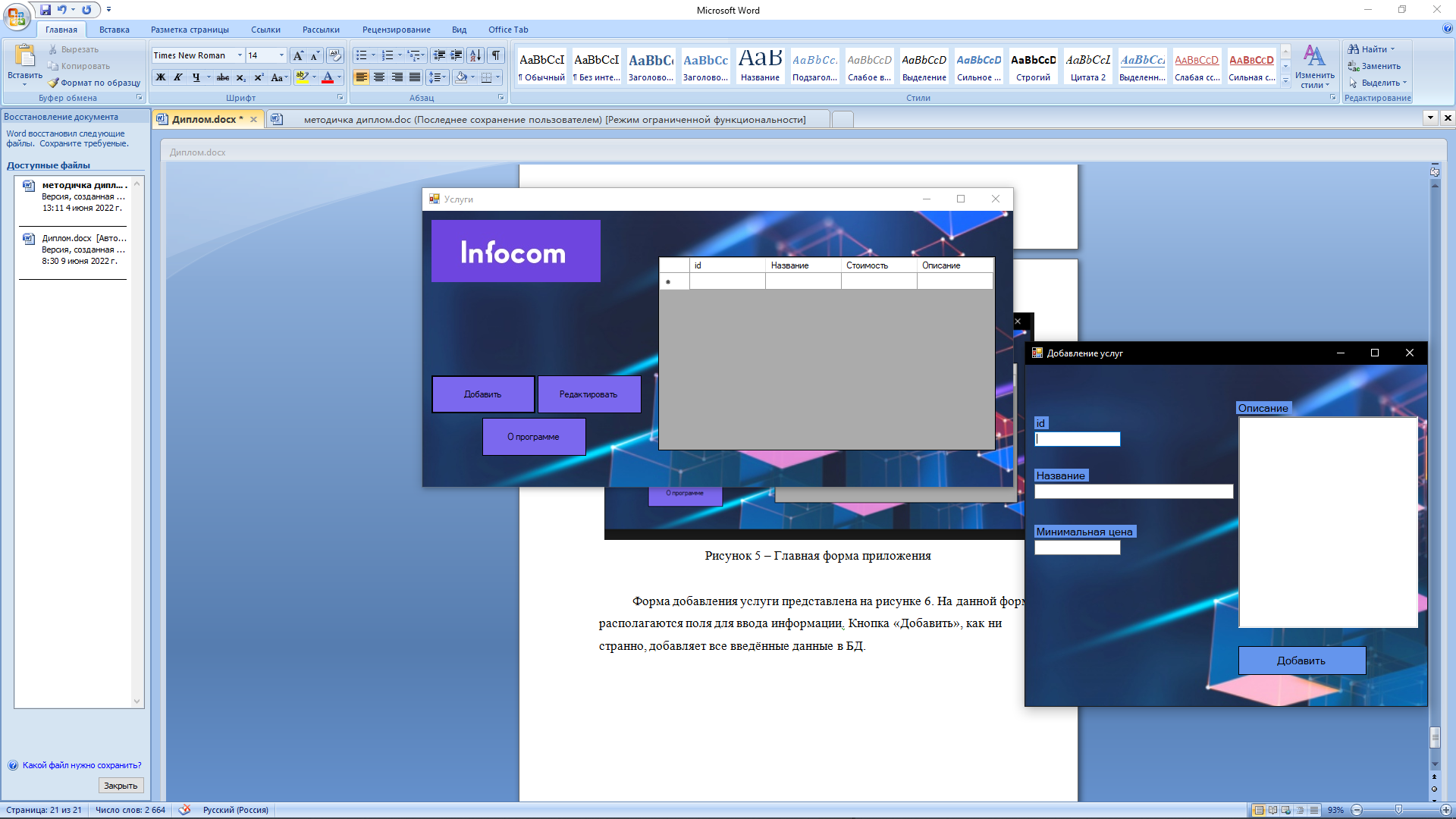


Рисунок 6 – Форма добавления новых услуг

Так же на главной форме имеется кнопка, которая выводит форму справки, на которой описано:

* Для кого было разработано;
* Для решения каких задач было разработано;
* Кем было разработано;

Форма справки располагается на рисунке 7.

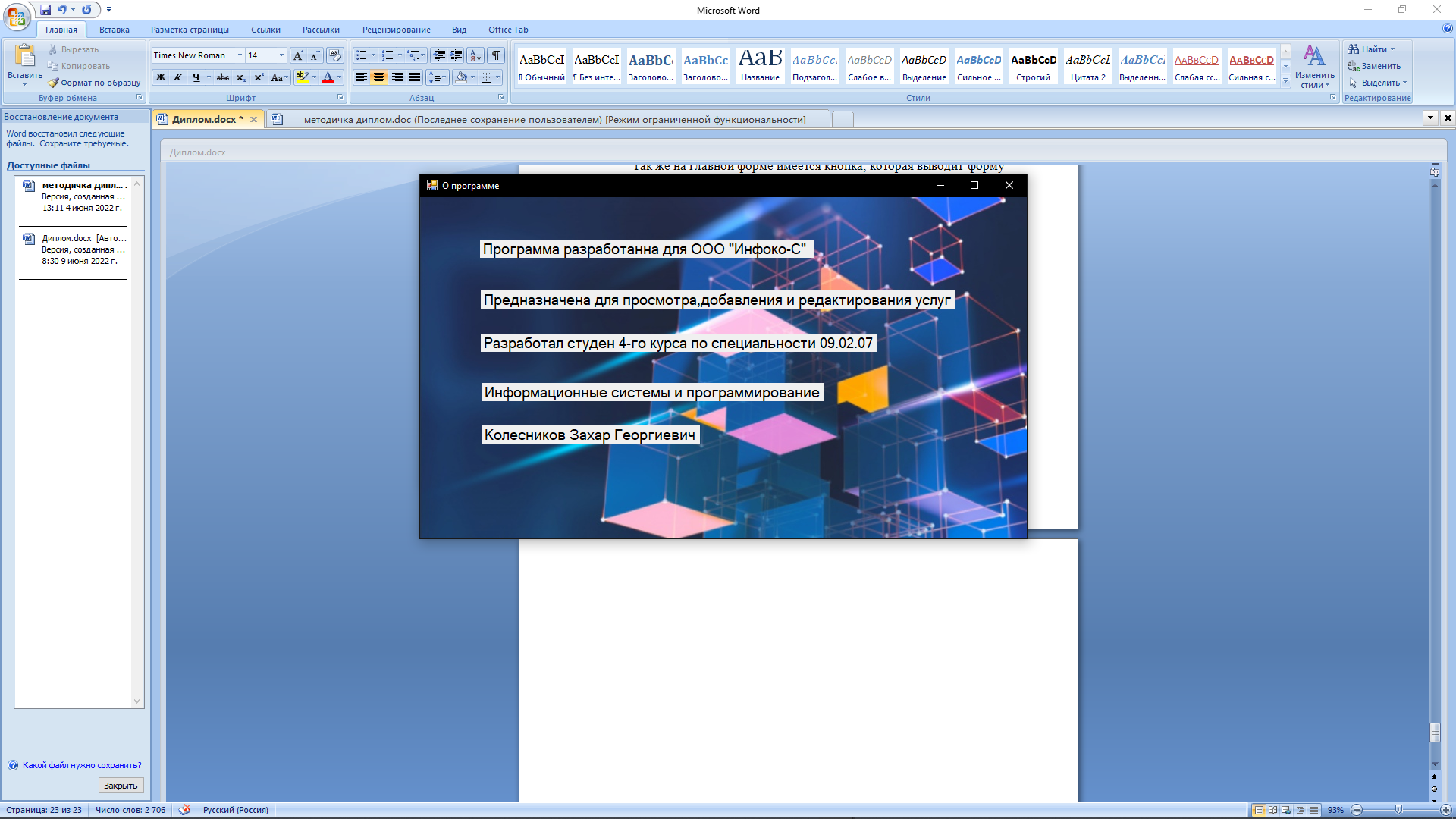


Рисунок 7 – Форма справки

Последняя форма – это форма редактирования имеющихся услуг. Она представлена на рисунке 8. На данной форме присутствуют поля для редактирования информации и поиска услуг по названию, так же присутствует кнопка, которая отправляет запрос на сервер для редактирования информации и кнопки для перехода на следующую или предыдущую запись.

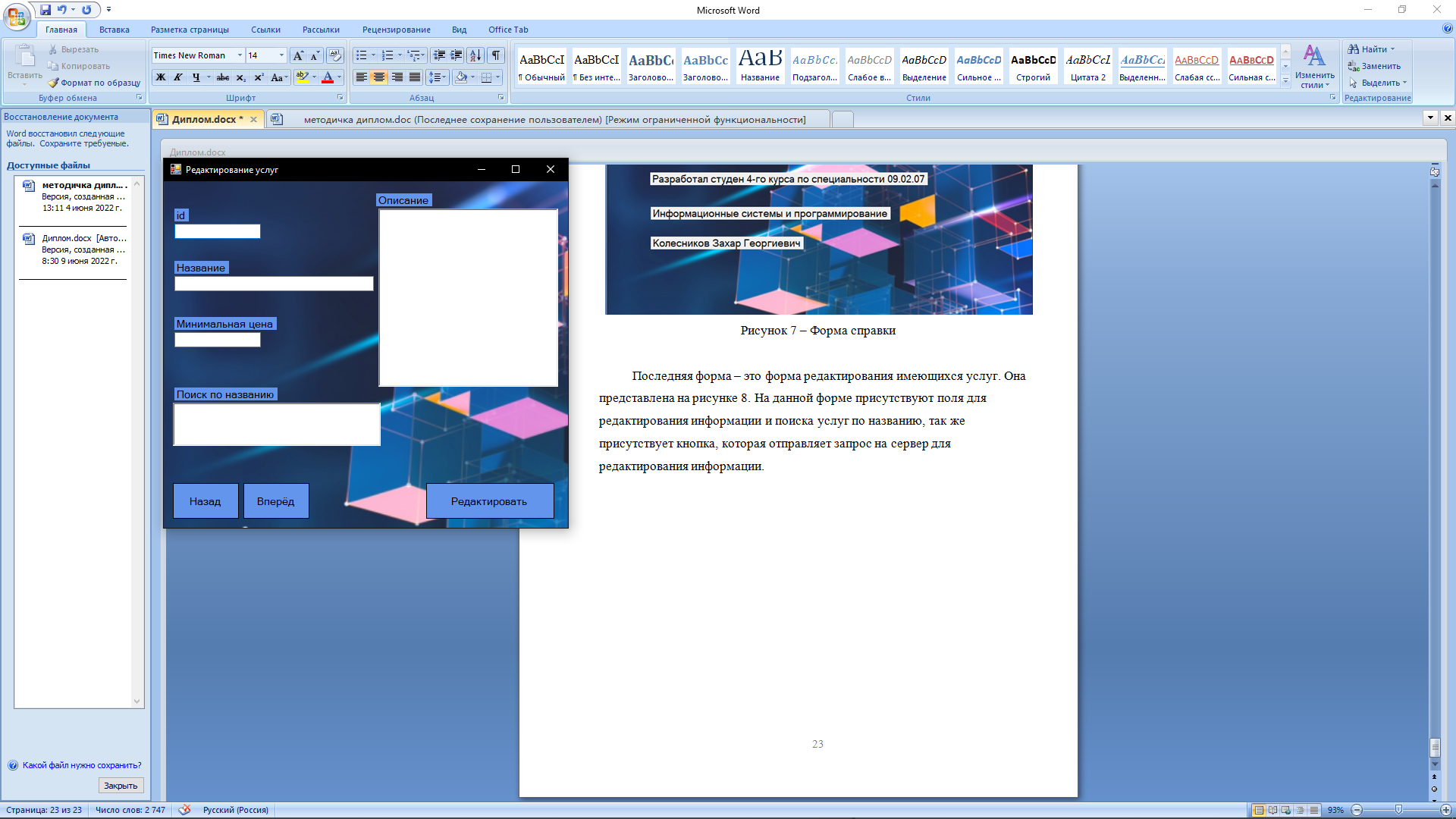


Рисунок 8 – Форма редактирования

На рисунке 9 приведено окно конструктора главной формы, который показывает с помощью какого способа загружаются данные в таблицу.

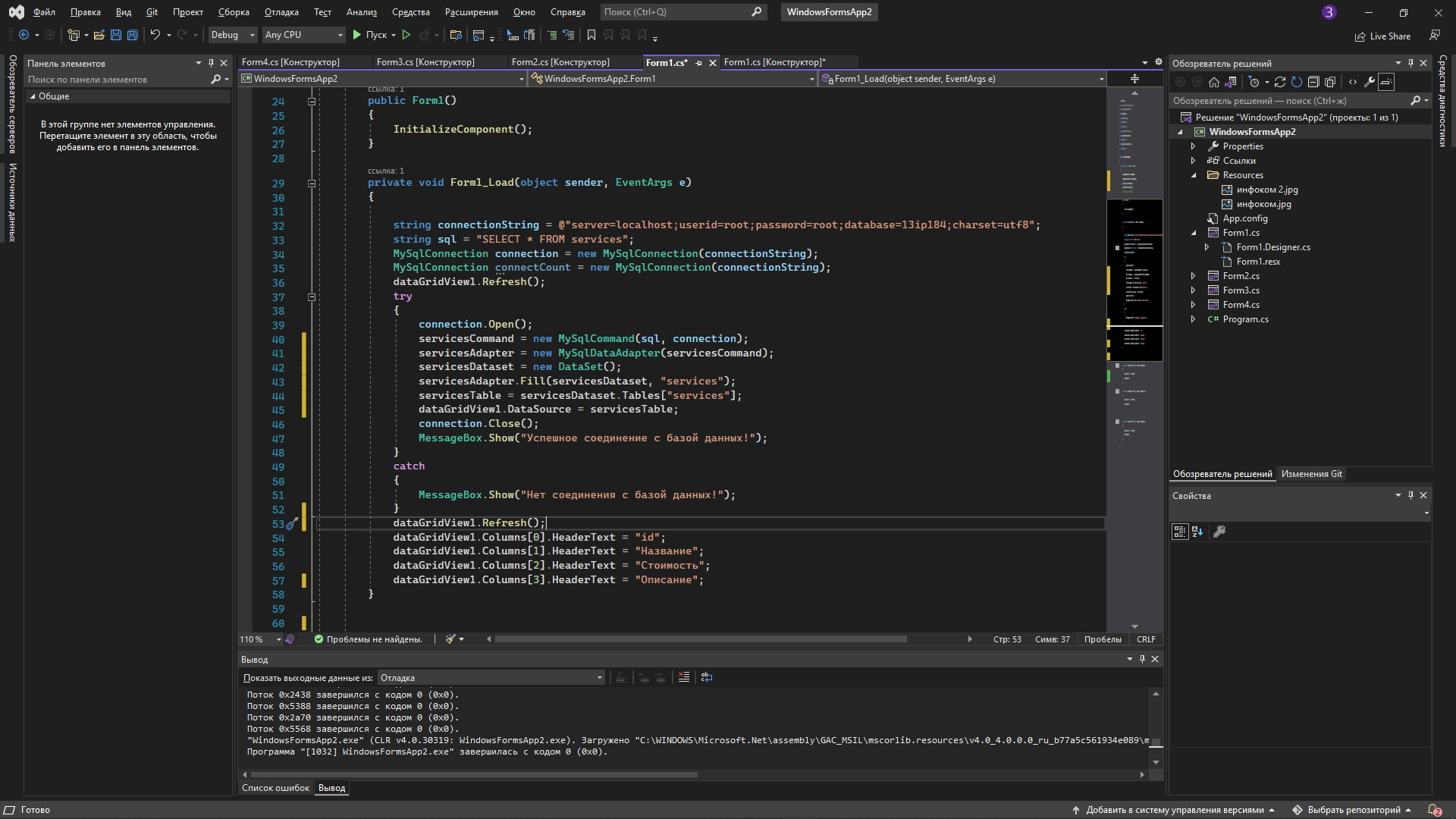


Рисунок 9 – Соединение и вывод информации

* 1. **Описание работы с сервером**

Серверную часть разрабатываемой информационной системы представляет MySQL-сервер баз данных. MySQL-служба, используемая для создания и управления базами данных на основе реляционной модели.

Информационная система подключается к серверу посредством NET–коннектора, после чего серверу отправляются SQL-запросы, при обработке которых и происходят вывод информации о услугах, добавление новых услуг в базу данных и редактирование имеющихся в базе данных услуг, а результат выполнения возвращает информационной системе.

Данная функция выполняется благодаря коду, который соединяет, открывает базу данных и заполняет таблицу в форме в графическом интерфейсе.

* 1. **Тестирование приложения**

Тестирование программного обеспечения – процесс проверки программного обеспечения с целью проверить соответствие между реальными результатами и ожидаемыми на конечном наборе тестов.

Цели тестирования:

1. Проверка выполнения указанных требований.

-У каждого продукта есть техническое задание. Тестировщики должны проверить все ли требования, описанные в ТЗ были выполнены.

2 Стать уверенным в уровне качества объекта тестирования.

-На прямую тестирование не влияет на качество продукта. Качество – это удовлетворение ожиданий пользователя, а удовлетворение зависит от большого разнообразия факторов.

Тем не менее, поиск и исправление ошибок позволяет продукту работать именно так, как и задумывалось в начале разработки.

Можно сказать, что если программа работает корректно и соответствует заданным условиям, то это говорит о достижения определённого уровня качества.

1. Предотвращение дефектов.

- Существует так же тестирование на более ранних этапах, тоже самое тестирование документации, ведь если заранее протестировать ТЗ, то тестировщик заранее может сообщить о предстоящих проблемах, которые могут возникнуть в результате разработки ПО. Зная про данные проблемы, можно заранее избежать их в будущем.

4 Обнаружение отказов.

- Поиск ошибок в программном обеспечении является неотъемлемой частью тестирования.

5 Предоставление заинтересованным лицам достаточной информации.

- Тестировщики не могут напрямую влиять на исправление дефектов, но могут показать текущее состояние продукта, выражая это в количестве ошибок и дефектов.

6 Снижение уровня риска.

- Чем лучше тестирование, тем меньше шанс пропустить критические ошибки, которые могут сильно повлиять на качество программного продукта, а значит и риск появления плохого качества программного обеспечения ниже.

**Приложение 1**

**Техническое задание на разработку программы "Услуга"**

**1. Введение**

**1.1. Наименование программы**

Наименование программы: "Услуга"

**1.2. Назначение и область применения**

Программа предназначена для управления содержимым базы данных, содержащим данные об улсугах

**2. Требования к программе**

**2.1. Требования к функциональным характеристикам**

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:  
2.1.1. Добавление данных в БД

2.1.2. Редактирование данных

2.1.3. Вывод данных

**2.2. Требования к надежности**

**2.2.1. Требования к обеспечению надежного функционирования программы**

Надежное (устойчивое) функционирование программы должно быть обеспечено выполнением Заказчиком совокупности организационно-технических мероприятий, перечень которых приведен ниже:   
а) организацией бесперебойного питания технических средств;   
б) использованием лицензионного программного обеспечения;   
в) регулярным выполнением рекомендаций Министерства труда и социального развития РФ, изложенных в Постановлении от 23 июля 1998 г. Об утверждении межотраслевых типовых норм времени на работы по сервисному обслуживанию ПЭВМ и оргтехники и сопровождению программных средств»;   
г) регулярным выполнением требований ГОСТ 51188-98. Защита информации. Испытания программных средств на наличие компьютерных вирусов

**2.2.2. Время восстановления после отказа**

Время восстановления после отказа, вызванного сбоем электропитания технических средств (иными внешними факторами), не фатальным сбоем (не крахом) операционной системы, не должно превышать 30-ти минут при условии соблюдения условий эксплуатации технических и программных средств.   
Время восстановления после отказа, вызванного неисправностью технических средств, фатальным сбоем (крахом) операционной системы, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.

**2.2.3. Отказы из-за некорректных действий пользователей системы**

Отказы программы вследствие некорректных действий пользователя при взаимодействии с программой через графический интерфейс недопустимы.

**3. Условия эксплуатации**

**3.1. Климатические условия эксплуатации**

Климатические условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям, предъявляемым к техническим средствам в части условий их эксплуатации

**3.2. Требования к квалификации и численности персонала**

Минимальное количество персонала, требуемого для работы программы, должно составлять не менее 2 штатных единиц — системный администратор и конечный пользователь программы — оператор. Системный администратор должен иметь высшее профильное образование и сертификаты компании-производителя операционной системы. В перечень задач, выполняемых системным администратором, должны входить:   
а) задача поддержания работоспособности технических средств;   
б) задачи установки (инсталляции) и поддержания работоспособности системных программных средств — операционной системы;   
в) задача установки (инсталляции) программы.   
г) задача создания резервных копий базы данных.

**3.3. Требования к составу и параметрам технических средств**

3.3.1. В состав технических средств должен входить IВМ-совместимый персональный компьютер (ПЭВМ), выполняющий роль сервера, включающий в себя:

3.3.1.1. процессор Pentium-2.0Hz, не менее;   
3.3.1.2. оперативную память объемом, 1Гигабайт, не менее;   
3.3.1.3. HDD, 5 Гигабайт, не менее;   
3.3.1.4. операционную систему Windows 2000 Server или Windows 7 профессиональная;  
3.3.1.6. Microsoft SQL Server 2000

**3.4. Требования к информационной и программной совместимости**

**3.4.1. Требования к информационным структурам и методам решения**

База данных работает под управлением Microsoft SQL Server. Используется много поточный доступ к базе данных. Необходимо обеспечить одновременную работу с программой с той же базой данной модулей экспорта внешних данных

**3.4.2. Требования к исходным кодам и языкам программирования**

Дополнительные требования не предъявляются.

**3.4.3. Требования к программным средствам, используемым программой**

Системные программные средства, используемые программой, должны быть представлены лицензионной локализованной версией операционной системы Windows 2000 Server или Windows 7 профессиональная и Microsoft SQL Server 2000.

**3.4.4. Требования к защите информации и программ**

Требования к защите информации и программ не предъявляются.

**3.5. Специальные требования**

Программа должна обеспечивать одновременную работу пользователей посредством графического интерфейса.

**4. Требования к программной документации**

**4.1. Предварительный состав программной документации**

Состав программной документации должен включать в себя:   
4.1.1. техническое задание;  
4.1.2. программу и методики испытаний;  
4.1.3. руководство оператора;

**5. Технико-экономические показатели**

**5.1. Экономические преимущества разработки**

Ориентировочная экономическая эффективность не рассчитываются. Аналогия не проводится ввиду уникальности предъявляемых требований к разработке.

**6. Стадии и этапы разработки**

**6.1. Стадии разработки**

Разработка должна быть проведена в три стадии:   
1. разработка технического задания;   
2. рабочее проектирование;   
3. внедрение.

**6.2. Этапы разработки**

На стадии разработки технического задания должен быть выполнен этап разработки, согласования и утверждения настоящего технического задания.   
На стадии рабочего проектирования должны быть выполнены перечисленные ниже этапы работ:

1. разработка программы;   
2. разработка программной документации;   
3. испытания программы.

На стадии внедрения должен быть выполнен этап разработки подготовка и передача программы.

**6.3. Содержание работ по этапам**

На этапе разработки технического задания должны быть выполнены перечисленные ниже работы:   
1. постановка задачи;   
2. определение и уточнение требований к техническим средствам;   
3. определение требований к программе;  
4. определение стадий, этапов и сроков разработки программы и документации на неё;   
5. согласование и утверждение технического задания.   
На этапе разработки программы должна быть выполнена работа по программированию (кодированию) и отладке программы.   
На этапе разработки программной документации должна быть выполнена разработка программных документов в соответствии с требованиями к составу документации.   
На этапе испытаний программы должны быть выполнены перечисленные ниже виды работ:   
1. разработка, согласование и утверждение и методики испытаний;   
2. проведение приемо-сдаточных испытаний;   
3. корректировка программы и программной документации по результатам испытаний.   
На этапе подготовки и передачи программы должна быть выполнена работа по подготовке и передаче программы и программной документации в эксплуатацию на объектах Заказчика.

**7. Порядок контроля и приемки**

**7.1. Виды испытаний**

Приемо-сдаточные испытания должны проводиться на объекте Заказчика в оговоренные сроки.   
Приемо-сдаточные испытания программы должны проводиться согласно разработанной Исполнителем и согласованной Заказчиком Программы и методик испытаний.   
Ход проведения приемо-сдаточных испытаний Заказчик и Исполнитель документируют в Протоколе проведения испытаний

**7.2. Общие требования к приемке работы**

На основании Протокола проведения испытаний Исполнитель совместно с Заказчиком подписывает Акт приемки-сдачи программы в эксплуатацию