|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  Калужский филиал  федерального государственного бюджетного  образовательного учреждения высшего образования  ***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»***  ***(КФ МГТУ им. Н.Э. Баумана)*** |

**ФАКУЛЬТЕТ** \_ ***ИУК «Информатика и управление»\_\_\_\_\_\_\_\_\_***

**КАФЕДРА** \_\_ ***ИУК5 «Системы обработки информации»***

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к курсовой работе на тему:**

***Разработка программы для обмена текстовыми сообщениями***

***с помощь Windows API***

по дисциплине ***Системное программирование***

Студент гр.\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

Оценка руководителя \_\_\_\_\_ баллов \_\_\_\_\_\_\_\_\_\_\_

30-50 (дата)

Оценка защиты \_\_\_\_\_ баллов \_\_\_\_\_\_\_\_\_\_\_

30-50 (дата)

Оценка проекта \_\_\_\_\_ баллов \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оценка по пятибалльной шкале)

Комиссия: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

Калуга, 2021

Калужский филиал   
федерального государственного бюджетного образовательного учреждения высшего образования

***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»   
(КФ МГТУ им. Н.Э. Баумана)***

УТВЕРЖДАЮ

Заведующий кафедрой **\_\_ИУК5\_\_\_**

\_\_\_\_\_\_\_\_\_\_\_(Е.В. Вершинин)

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_г.

**ЗАДАНИЕ**

**на выполнение курсовой работы**

по дисциплине ***Системное программирование***

Студент ***Мицих Владислав Павлович ИУК5-42Б*** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилия, инициалы, индекс группы)

Руководитель ***Фролов Павел Валерьевич*** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилия, инициалы)

График выполнения проекта: 25% к\_4\_нед., 50% к\_7\_нед., 75% к\_10\_нед., 100% к\_14\_нед.

***1. Тема курсового проекта***

***Разработка программы для обмена текстовыми сообщенями***

***при помощи Windows API***

***2. Техническое задание***

*Разработать приложение с использованием функций Windows API для обмена текстовыми сообщениями*

***3. Оформление курсового проекта***

3.1. Расчетно-пояснительная записка на\_\_\_\_\_\_\_\_ листах формата А4.

3.2. Перечень графического материала КП (плакаты, схемы, чертежи и т.п.)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

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

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

Дата выдачи задания «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_г.

Руководитель курсового проекта \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

(подпись) (Ф.И.О.)

Задание получил\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_20\_\_г.

(подпись) (Ф.И.О.)

Примечание:

Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

**СОДЕРЖАНИЕ**

1. Техническое задание…………………………………………………………...5-9
   1. Наименование……………………………………………………………...5
   2. Основание для разработки………………………………………………...5
   3. Исполнитель………………………………………………………………..6
   4. Цель разработки……………………………………………………………6
   5. Содержание работы………………………………………………………..7
      1. Задачи, подлежащие решению………………………………………...7
      2. Требования к архитектуре АСОИ……………………………………...7
      3. Требования к составу программных компонентов…………………...7
      4. Требования к прикладным программам………………………………7
      5. Требования к входным/выходным данным…………………………...8
      6. Требования к временных характеристикам…………………………..8
      7. Требования к составу технических средств…………………………..8
   6. Этапы разработки……………………..…………………………………...9
   7. Техническая документация, предъявляемая по окончании работы……9
   8. Дополнительные условия……………………..…………………………..9
2. Научно-исследовательская часть…………………………………………...10-19
   1. Постановка задачи проектирования……………………………………..10
   2. Описание предметной области…………………………………………..10
   3. Анализ аналогов и прототипов……………………..……………………12
   4. Перечень задач, подлежащих решению в процессе разработки……….17
   5. Обоснование инструментов и платформы для разработки……............18
3. Проектно-конструкторская часть…………………………………………..22-32
   1. Разработка структуры приложения……………………………………...20
   2. Используемые функции Win32API……………………………………...20
   3. Разработка архитектуры приложения…………………………………...25
   4. Разработка и реализация структуры приложения………………………25
   5. Разработка алгоритмов приложения…………………………………….26
   6. Разработка интерфейса взаимодействия пользователя системой…………………………………………………………………..31
4. Проектно-технологическая часть…………………………………………..33-36
   1. Разработка руководства пользователя………….……………………….33
   2. Установка и запуск.… …………………………………………………….35
   3. Тестирование и отладка макета рабочей программы…………………...38

5. Заключение…………………………………..…………………………………..39

6. Список использованной литературы…………………………………………..40

# 1.ТЕХНИЧЕСКОТЕХНИЧЕСКОЕ ЗАДАНИЕ

**1.1. Наименование**

Чат на основе Windows API.

**1.2. Основание для разработки**

Системы обмена сообщениями — самые доступные и востребованные средства общения в Интернете, а также в корпоративных и локальных сетях. Эти службы делятся на две большие группы:

службы обмена сообщениями в режиме офлайн (почтовые системы e-mail);

службы мгновенных сообщений в режиме онлайн (Instant Messaging Service, IMS).

Чат - средство обмена сообщениями по компьютерной сети в режиме реального времени, а также программное обеспечение, позволяющее организовывать такое общение. Характерной особенностью является коммуникация именно в реальном времени или близкая к этому, что отличает чат от форумов и других «медленных» средств. То есть, если на форуме можно написать вопрос и ждать, пока кто-нибудь посчитает нужным на него ответить (в то же время, можно получить и несколько ответов сразу от разных пользователей), то в чате общение происходит только с теми, кто присутствует в нём в настоящий момент, а результаты обмена сообщениями могут и не сохраняться.

Под словом чат обычно понимается групповое общение, хотя к ним можно отнести и обмен текстом «один на один» посредством программ мгновенного обмена сообщениями, например, XMPP, ICQ или даже SMS.

Сетевой чат может быть использован как для малого круга людей, т.е. в домашних условиях, так и на производстве небольшого предприятия.

**1.3. Исполнитель**

Студент группы ИУК5-42Б Мицих Владислав Павлович

**1.4. Цель разработки**

Целью курсовой работы является формирование практических навыков по разработке и реализации программного приложения с использованием интерфейса прикладного программирования (АРI) операционных систем.

Задачи проектирования:

1. овладение первичными навыками ведения научно-исследовательской, проектной и производственно-технологической деятельности, развитие творческих способностей индивидуально для каждого студента;
2. подготовка к выполнению выпускной квалификационной работы;
3. усвоение методов грамотного ведения, оформления и редактирования технической документации.

Целью разработки является создание программы, которая помогает обмениваться мгновенными текстовыми сообщениями.

**1.5. Содержание работы**

**1.5.1. Задачи, подлежащие решению:**

исследование существующих протоколов передачи данных, а также изучение работы сокетов;

изучение принципов системного программирования и работы WinAPI функций;

реализация приложения для обмена сообщениями на основе WinAPI;

1. тестирование разработанного программного обеспечения;
2. подготовка расчетно-пояснительной записки и графических листов;
3. подготовка презентации и речи для защиты курсовой работы;
4. защита курсовой работы.

**1.5.2. Требования к архитектуре АСОИ**

К архитектуре предъявляются следующие требования:

1. тип приложения – оконное;

2. клиент – серверная архитектура.

**1.5.3. Требования к составу программных компонентов**

Программный комплекс должен состоять из следующих программных компонентов:

1. исполняемый файл PE формата с расширением .exe, реализующий клиентскую часть чата.

**1.5.4. Требования к прикладным программам**

Для работы программного комплекса необходимы:

1. Microsoft Windows;
2. Microsoft Visual Studio;
3. минимальный набор драйверов, обеспечивающих   
   работоспособность ПК.

**1.5.5. Требования к входным/выходным данным**

1. Входные данные:
2. собственный порт в диапазоне от 1024 до 49151;
3. порт собеседника в диапазоне от 1024 до 49151;
4. IP – адрес версии IPv4;
5. «ник-нейм» пользователя;
6. текстовые сообщения пользователей.

Выходные данные:

1. всплывающее окно типа «Контакт», «Сообщения»;
2. список контактов в текстовом формате для экспорта данных.

**1.5.6. Требования к временным характеристикам**

Требования к временным характеристикам программы не предъявляются.

**1.5.7. Требования к составу технических средств**

Для функционирования системы необходимы:

процессор: 1 ГГц и быстрее;

* сетевой адаптер;
* RAM: 1 Гбайт (32 бит) или 2 Гбайт (64 бит);
* HDD: 500 Мбайт (32 бит) или 700 Мбайт (64 бит);
* видеокарта: поддержка Microsoft DirectX 9 с драйвером WDDM;
* монитор;
* мышь;
* клавиатура.

**1.6. Этапы разработки**

создание пользовательского интерфейса с помощью WinAPI;

1. тестирование реализованной технологии;
2. исправление выявленных ошибок.

**1.7. Техническая документация, предъявляемая по окончании работы**

По окончанию работы предъявлена расчетно-пояснительная записка в состав которой входят:

* техническое задание;
* научно-исследовательская часть;
* проектно-конструкторская часть;
* проектно-технологическая часть.

Также должна быть предоставлена графическая часть работы, выполненная формате А1 на 2 листах, в которую входят:

* демонстрационные чертежи;
* алгоритмические схемы.

**1.8. Дополнительные условия**

язык программирования С++;

использование Windows API функций;

1. интерфейс должен предоставлять список контактов пользователя, окно для ввода и получения сообщений, форма настройками клиента.

**Научно-исследовательская часть**

**2.1. Постановка задачи проектирования** Задача курсовой работы состоит в разработке децентрализованного peer-to-peer чата на основе Windows API, где сообщения передаются с одного компьютера на другой.

**2.2. Описание предметной области**

Чат – средство обмена сообщениями в реальном времени с помощью компьютера, смартфона, планшета в основном через интернет. Чат появился в результате развития информационных технологий. Ранней альтернативой чату был телефон, который позволял общаться в режиме реального времени.

Существует несколько разновидностей программной реализации чатов: HTTP или веб-чаты. Такой чат выглядит как обычная веб-страница, где можно прочесть последние несколько десятков фраз, написанные участниками чата и модераторами. Страница чата автоматически обновляется с заданной периодичностью.

Чаты, использующие технологию Adobe Flash. Вместо периодической перезагрузки страницы между клиентом и сервером открывается сокет, что позволяет моментально отправлять или получать сообщения, расходуя меньше трафика.

IRC, специализированный протокол для чатов.

Программы-чаты для общения в локальных сетях (например, Vypress Chat, Intranet Chat, Pichat). Часто есть возможность передачи файлов.

Чаты, реализованные поверх сторонних протоколов (например, чат, использующий ICQ).

Чаты, работающие по схеме клиент-сервер, это позволяет использовать их в сетях со сложной конфигурацией, а также управлять клиентскими приложениями (например, Mychat, Jabber). Чаты, работающие в одноранговых сетях. У них нет потребности в отдельном сервере, они часто используют возможности технологий DHT и TCP Relay (пример: Tox). Чаты, использующие технологию Push. Вместо периодической отправки запросов серверу о новых сообщениях, используются входящие сообщения от сервера, что позволяет отправлять и получать сообщения, расходуя меньше трафика (например, WinGeoChat)

Полностью анонимные чаты. В них собеседник не предполагает с кем общается и при каждом новом соединении общается с новым человеком. (например, ChatVdvoem (ЧатВдвоем))

Среди плюсов систем обмена мгновенными сообщениями можно отметить следующие:

Экономия времени. Одно сообщение с подробным описанием того, что и как нужно сделать, и картинками, стоит целого телефонного звонка. И это сообщение не потребует повторных звонков с уточнениями, ведь оно хранится на компьютере, в истории принятых сообщений.

Работа с группой. Возможно мгновенно информировать целую группу людей простым оповещением в соответствующий чат.

Всегда точная доставка до адресата. Отправленные сообщения хранятся на сервере и будут доставлены получателю, даже если его мобильное устройство отключено в текущий момент времени. Достаточно только включить свой компьютер и получить их. А если потребуется прочитать свою историю сообщений в дороге, получатель сможет это сделать через обычный WEB-браузер, через Интернет.

Снижение затрат на связь. Обмен сообщениями позволяет значительно снизить затраты на междугороднюю и сотовую связь. Ведь для этого будет использоваться уже оплаченный интернет-канал.

Безопасность. Все сообщения надежно шифруются, что делает общение безопасным.

При всех существенных преимуществах также следует отметить и слабые стороны, такие как:

Зависимость от качества связи. Эти службы более зависимы от хорошей работы интернета или другой сети. Если сеть не работает или работает слишком медленно, обмен мгновенными сообщениями использовать нельзя. С качеством связи также связан следующий пункт.

Искажение сообщений. Если человек находится в движении, с мобильным устройством, и на его пути встречаются зоны с плохой связью, сообщение может прийти искаженным, с опозданием или его передача в итоге может вообще не завершиться.

Неудобство работы. Часто уведомления о сообщениях могут быть скорее надоедливыми и мешать пользователю работать с основной системой. А во время работы нескольких программных продуктов небольшие мигающие уведомления о сообщениях могут даже не быть видны на экране.

Возможные сбои. Некоторые приложения могут зависнуть или дать сбой во время приема или отправки сообщения. В итоге служба может быть не в состоянии вернуться к операции, которую она выполняла до получения сообщения.

**2.3. Анализ аналогов и прототипов**

**ICQ** (I seek you — «я ищу тебя») — бесплатная система мгновенного обмена текстовыми сообщениями для мобильных и иных платформ с поддержкой голосовой и видеосвязи. Позволяет пересылать текстовые сообщения, изображения, видео и аудио через Интернет.

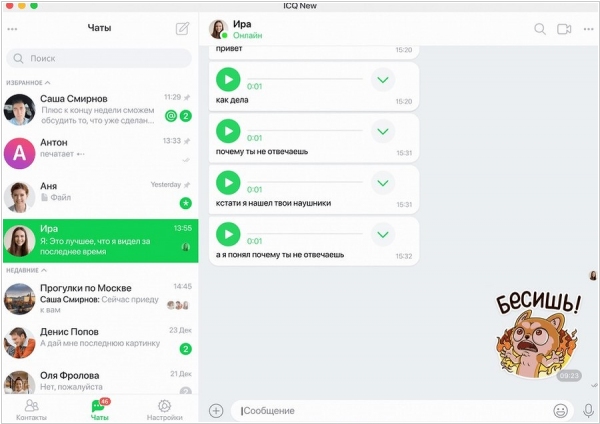


Рисунок 1 - Интерфейс программы «ICQ New»

Для использования службы требуется зарегистрировать учётную запись, что может быть сделано через интерфейс клиента, а также интернет-портал. Для идентификации пользователей служба использует UIN (Universal Identification Number) — уникальный для каждой учётной записи номер, состоящий из 5-9 арабских цифр. Этот номер присваивается учётной записи при первичной регистрации пользователя в системе, после чего, в паре с паролем, может использоваться для аутентификации в системе.



Рисунок 2 – Регистрация в программе

История сообщений на серверах не сохраняется, и может храниться только локально программой-клиентом на устройстве пользователя, и быть доступной через её интерфейс.

Инициировать переписку с другим пользователем и добавить его в список контактов можно, зная его UIN. В свою очередь, для поиска пользователей в системе существует внутренняя функция поиска, доступная через интерфейс клиента, которая позволяет получать списки пользователей, удовлетворяющих введённым в поисковом запросе критериям: совпадению по никнейму, а также дополнительной введённой пользователями информации.

С каждой учётной записью ассоциирован статус присутствия, являющийся индикатором того, подключён пользователь к сети или нет, и готов ли он в данный момент отвечать на сообщения. В списке контактов и в окне диалога показывается также статус пользователя.

Основной статус пользователя служит индикатором его присутствия в системе и готовности отвечать на сообщения (хотя в последнее время число статусов резко возросло и перестало отвечать этим требованиям). Традиционно существует ряд основных статусов.

Если пользователь вообще отсутствует в системе на данный момент (не авторизован), то ему присваивается статус «Не в сети» или «Офлайн»; если же он в сети (авторизован), то ему присваивается статус «В сети» или «Онлайн».

Также существуют несколько промежуточных статусов:

* Отошёл  — свидетельствует о том, что пользователь в течение определённого времени не проявлял активности (не двигал мышью и не нажимал клавиш). Статус может ставиться и вручную, обычно им сигнализируют небольшой период отсутствия за компьютером.
* Недоступен — говорит о том, что пользователь получил статус «Отошёл» и с этого момента в течение определённого времени продолжал пребывать в неактивном состоянии. Также может быть выставлено принудительно и обычно им сигнализируют долгосрочное отсутствие за компьютером (исключение: в ICQ 5.x этот статус выставляется автоматически при переходе в полноэкранный режим).

Как правило, автопереход в эти статусы можно отключить в настройках клиента.

Существуют также статусы «Свободен для разговора, «Не беспокоить» и «Занят». Начиная с ICQ 4, их нельзя проставлять, но можно считывать у других пользователей, которые, например, используют ICQ 2003b или ниже, а также у многих альтернативных клиентов.

Помимо этого, для учётной записи пользователь может устанавливать особый режим — «режим невидимости». В этом режиме другие пользователи будут получать для учётной записи статус присутствия «не в сети», а знать об активном режиме невидимости и получать актуальный статус присутствия смогут только пользователи, чьи учётные записи были внесены в список видимости пользователя.

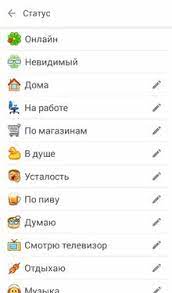


Рисунок 3 – Список статусов

До выхода ICQ 2001 список контактов хранился на стороне клиента. Начиная с ICQ 2001, контакты пользователя хранятся на сервере, что позволяет не добавлять контакты повторно при входе в свою учётную запись с разных устройств. Контакты в списке могут быть разделены на группы, имена и количество которых изменяются пользователем.

При добавлении контакта может потребоваться авторизация — разрешение видеть его статус присутствия и отправлять ему файлы. Для таких контактов формируется запрос на авторизацию, который доходит до пользователя добавляемой учётной записи в виде системного уведомления, на которое он может ответить либо согласием, либо отказом, по желанию снабдив решение текстовым комментарием. Максимально можно иметь 1000 контактов.

Для обеспечения необходимого уровня конфиденциальности в ICQ существует несколько списков, выполняющих определённую функцию, в которые каждый пользователь может заносить любые контакты без уведомления последних.

Существует 3 списка:

* Список игнорируемых — от оказавшихся в этом списке пользователей не приходит никаких уведомлений, им не показывается никакой статус добавившего, кроме «Не в сети». При добавлении в этот список пользователь удаляется из списка контактов, при удалении добавляется автоматически. Если контакт требует авторизации, то её придётся запрашивать снова.
* Список видящих — пользователям из этого списка показывается статус «Невидимый», если он выбран, за исключением опции «Невидимый для всех», возможной в некоторых альтернативных клиентах (например QIP, Miranda, RnQ и прочие).
* Список невидящих — пользователям из этого списка всегда показывается статус «Не в сети», за исключением опции «Видимый для всех», возможной в некоторых альтернативных клиентах.

Максимальное количество контактов в каждом списке ограничено.

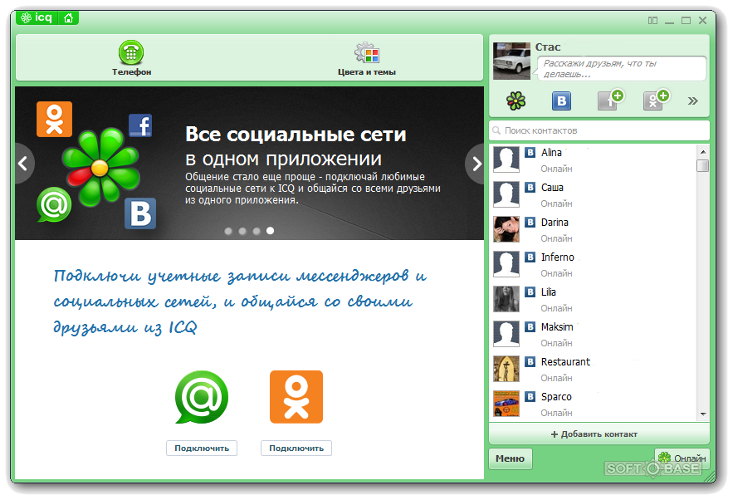


Рисунок 4 – Список контактов

С каждым из контактов можно вести личную переписку. Если отправитель не отключил эту возможность, то, в зависимости от клиента, получатель информируется о наборе сообщения, что создаёт эффект присутствия отправителя. Длина отправляемого сообщения ограничена.

В случае, если в момент отправки сообщений адресат не находился в сети, они будут сохранены службой и доставлены адресату, как только тот подключится к сети.

Служба позволяет использовать в тексте отправляемых сообщений форматирование, а именно — определять гарнитуру, цвет, начертание и размер шрифта фрагментов сообщения, направление текста, а также вставлять в сообщения смайлы.

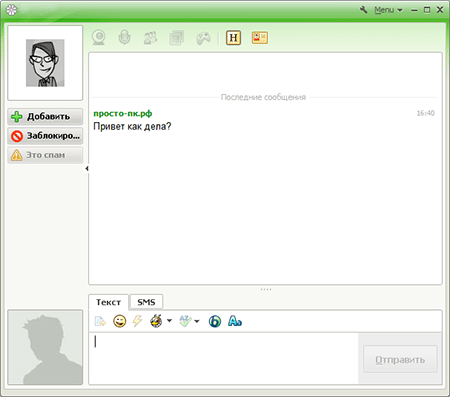


Рисунок 5 – Диалог в программе «ICQ»

**MyChat -** клиент-серверный чат, идеально подходит для передачи сообщений и обмена файлами в офисе или компании, может работать внутри корпоративной сети, а также через Интернет. Будет полезен компаниям, которым нужно средство корпоративного общения, где недопустимо использование таких публичных IM систем, как ICQ или Skype, в связи с ограниченным доступом к Интернету или корпоративными требованиями к безопасности для предотвращения утечки информации.

Клиент позволяет подключаться к отдельным каналам, производить регистрацию в каналах, вести приватные разговоры, передавать файлы, просматривать доски объявлений, искать пользователей, создавать оповещения с различными параметрами.

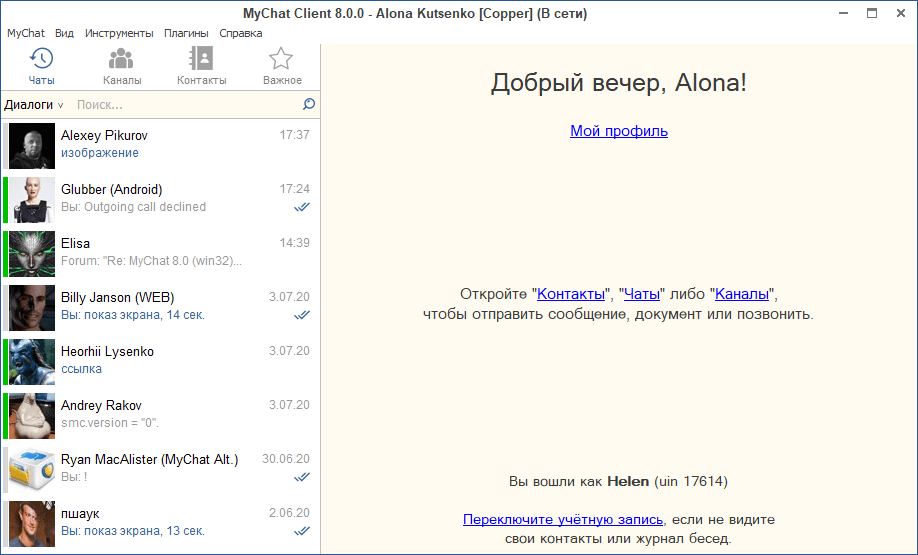


Рисунок 6 - Интерфейс программы «MyChat»

В силу того, что в оповещениях нельзя вести диалог, как в обычном приватном разговоре, существует отправка многоадресных приватных сообщений. Выделяете несколько людей в списке контактов мышкой с зажатой клавишей Ctrl или Shift и отправляете приватное сообщение:

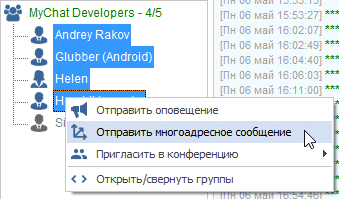


Рисунок 7 - Процесс отправления многоадресных сообщений

Все отправленные сообщения попадают к в историю, но диалоги не открываются, чтобы не плодить лишние окошки. Если придёт ответ, программа сразу это покажет.

Для частых однотипных отправок можно пользоваться собственными уже готовыми списками людей, как в оповещениях:

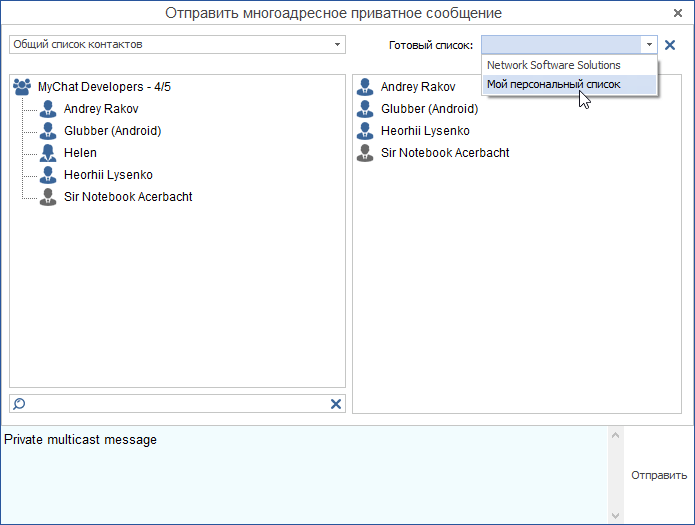


Рисунок 8 - Списки контактов клиента

**2.4. Перечень задач подлежащих решению в процессе разработки**

1. Должны быть реализованы стандартные функции работы чата: задание псевдонима, ввод портов для подключения, отправка и получение сообщений.

2. Программа должна позволять пользователю менять ряд настроек. В частности, имя пользователя в чате, шрифт и размер шрифта текстового поля, а также должна быть реализована возможность сохранения логфайлов.

3. Программный продукт должен иметь дружественный интерфейс, не требующий специальной подготовки пользователей, должен способствовать более эффективному распределению и экономии времени.

**2.5. Обоснование выбора инструментов и платформы для разработки**

Платформой для разработки была выбрана Windows 10, которая на данный момент является самой распространенной ОС в мире в соответствии с исследованием от NetMarketShare. Статистика приведена на рисунке 5

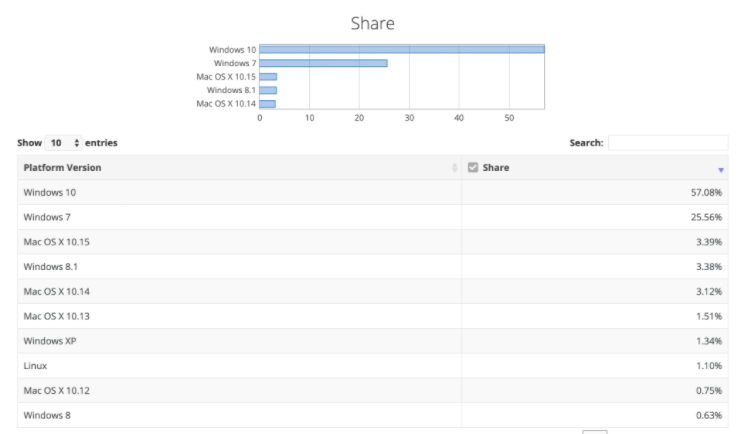


Рисунок 9 - Статистика использования операционных систем в мире.

Windows - семейство проприетарных операционных систем корпорации Microsoft, ориентированных на применение графического интерфейса при управлении.

Windows представляет гибкий инструмент в виде библиотек и функций, основные платформы приложений каждая из которых имеет ряд преимуществ:

* Универсальная платформа Windows(UWP)-предоставляется система общих типов, интерфейсы API и модель приложений для всех устройств под управление Windows 10;
* Win32/Win 64 – исходные платформы для нативных Windows приложений C/C++, которым требуется прямой доступ к Windows и оборудованию.

В качестве языка программирования был выбран язык С++. Язык успешно позволяет создать как простые приложения и утилиты, так и структурно сложные проекты. С++ является ISO-стандартизированным языком программирования

C++ - компилируемый статически типизированный язык программирования общего назначения. Поддерживая разные парадигмы программирования, сочетает свойства как высокоуровневых, так и низкоуровневых языков. Позволяет работать с библиотеками. Область его применения включает операционные системы, прикладные программы, драйверы устройств, приложения для встраиваемых систем, высокопроизводительные серверы, а также развлекательные приложения.

Средой проектирования, позволяющей реализовать подобное программное обеспечение, была выбрана Microsoft Visual Studio 2019 представлена на рисунке 6. Среда разработки Visual Studio представляет собой полный набор средств для разработки и создания веб приложений, настольных приложений и мобильных приложений.Visual C++ использует единую интегрированную среду разработки (IDE), которая позволяет совместно использовать средства и упрощает создание решений на базе нескольких языков.

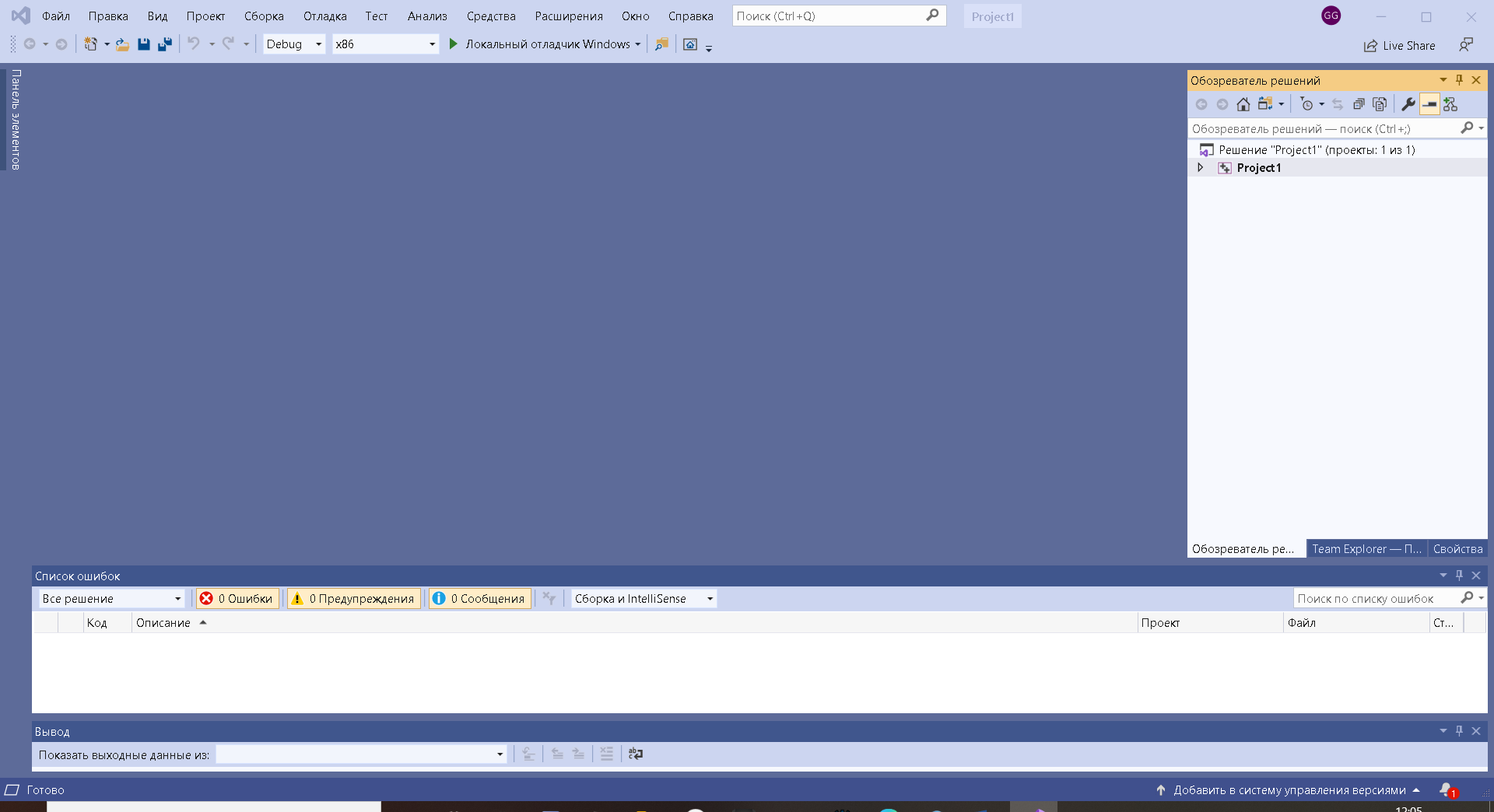


Рисунок 10 -Интерфейс Microsoft Visual Studio 2019.

Преимущества выбранной среды программирования:

* удобство интерфейса;
* сравнительно не высокие требования к системе;
* удобное средства отладки;
* возможность хранения данных в облаке;
* поддержка множества языков программирования;
* наличие бесплатной редакции Community;
* встроенный контроль за выполнением многопоточного кода.

# 3. Проектно-конструкторская часть

## 

## **3.1 Разработка структуры приложения**

Для разработки приложения использовалась среда Microsoft Visual Studio, включающая в себя широкий набор функциональных инструментов.

Ввод данных в приложении осуществляется через графический интерфейс с использованием клавиатуры. При этом вводимые данные могут быть на любом языке с использованием чисел и системных символов.

После входа в приложение пользователь должен ввести своё имя или псевдоним, в соответствующее текстовое поле, а также собственный порт и порт собеседника, после чего может начать общение в чате. Ввод обоих портов является обязательным, так как наличия неадресуемого пользователя в чате недопустимо.

Написав сообщение его нужно послать, при помощи специальной кнопки отправить. Следствием правильной работы приложения является отображение отправленного сообщения в окне диалога.

Общее назначение программного средства - общение по средствам отправки текстовых сообщений.

**3.2 Используемые функции Win32API**

**wWinMain -** вызывается системой как начальная точка входа, для базирующейся на Windows, прикладной программы.

int WINAPI wWinMain(

HINSTANCE hInstance, // дескриптор текущего экземпляра окна

HINSTANCE hPrevInstance, // дескриптор предыдущего экземпляра окна

LPSTR lpCmdLine, // указатель на командную строку

int nCmdShow // показывает состояние окна

);

**DefWindowProc** - вызывается оконной процедурой по умолчанию, чтобы обеспечить обработку по умолчанию любого сообщения окна, которые приложение не обрабатывает. Эта функция гарантирует то, что обрабатывается каждое сообщение.

DefWindowProc(

HWND hWnd, // Дескриптор процедуры, которая получает сообщение

UINT Msg, // Определяет сообщение

WPARAM wParam,// Определяет дополнительную информацию о сообщении

LPARAM lParam // Определяет дополнительную информацию о сообщении

);

**CreateWindowEx** - создает перекрывающее, выпрыгивающее или дочернее окно с расширенным стилем.

CreateWindowEx(

DWORD dwExStyle,      // улучшенный стиль окна  
LPCTSTR lpClassName,  // указатель на зарегистрированное имя класса  
LPCTSTR lpWindowName, // указатель на имя окна  
DWORD dwStyle,        // стиль окна  
int x,                // горизонтальная начальная позиция окна  
int y,                // вертикальная начальная позиция окна  
int nWidth,           // ширина окна  
int nHeight,          // высота окна  
HWND hWndParent,      // дескриптор родительского или окна владельца  
HMENU hMenu,          // дескриптор меню или ID дочернего окна  
HINSTANCE hInstance,  // дескриптор экземпляра прикладной программы  
LPVOID lpParam        // указатель на данные создания окна

);

**GetWindowText -** копирует текст заголовка определяемого окна (если окно имеет его) в буфер. Если заданное окно является органом управления, копируется его текст.

GetWindowText(

HWND hWnd, // дескриптор окна или элемента управления с текстом

LPTSTR lpString, // адрес буфера для текста

int nMaxCount // максимальное число символов для копирования

);

**TranslateMessage -** переводит сообщения виртуальных клавиш в символьные сообщения. Символьные сообщения помещаются в очереди сообщений вызывающего потока для прочтения.

TranslateMessage(

const MSG\* lpMsg // указатель на структуру MSG

);

**ShowWindow -** устанавливает состояние показа определяемого окна.

ShowWindow(

HWND hWnd, // дескриптор окна

int nCmdShow // состояние показа окна

);

**RegisterClass -** регистрирует класс окна для последующего использования при вызове функции CreateWindow или CreateWindowEx

RegisterClass(

CONST WNDCLASS\* lpWndClass // указатель на структуру WNDCLASS

);

**DispatchMessage -**  распределяет сообщение оконной процедуре. Обычно она используется, чтобы доставить сообщение, извлеченное функцией GetMessage.

DispatchMessage(

const MSG\* lpmsg // указатель на структуру MSG, которая содержит сообщение.

);

**SendMessage -**  отправляет заданное сообщение окну или окнам. Функция вызывает оконную процедуру для заданного окна и не возвращает значение до тех пор, пока оконная процедура не обработает сообщение.

SendMessage(

HWND hWnd, // дескриптор окна, процедура принимает сообщение

UINT Msg, // определяет сообщение, которое будет отправлено

WPARAM wParam, // определяет дополнительную информацию

LPARAM lParam // определяет дополнительную информацию

);

## **3.3 Разработка архитектуры приложения**

Архитектура приложения - клиент серверная, она представляет собой иерархическую сеть, которая состоит из узлов-клиентов (их может быть от одного и до неограниченного количества) и центрального сервера, через который выполняется хранение и обработка данных, а также передача их в обоих направлениях.

Клиент - это объект, запрашивающий информацию по сети. Как правило, это персональный компьютер или рабочая станция, запрашивающая информацию у сервера.

Сервер - компьютер, хранящий и обрабатывающий информацию, с которой работают сетевые клиенты.

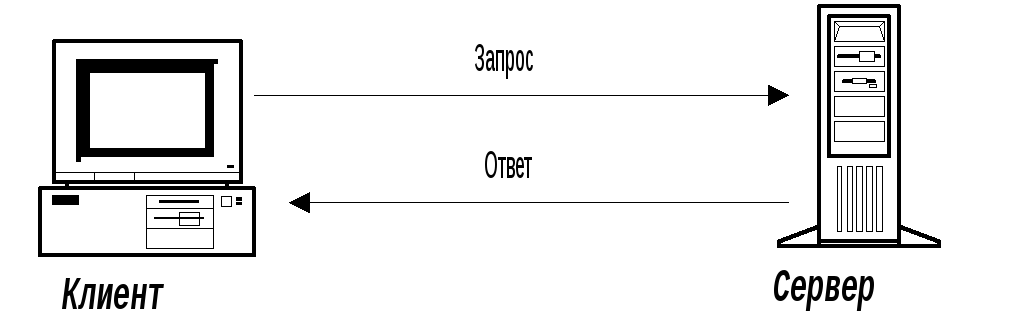


Рисунок 11 - Принцип работы клиент – серверной архитектуры

UDP — User Datagram Protocol. UDP — очень быстрый протокол, поскольку в нем определен самый минимальный механизм, необходимый для передачи данных. Конечно, он имеет некоторые недостатки. Сообщения поступают в любом порядке, и то, которое отправлено первым, может быть получено последним.

Доставка сообщений UDP вовсе не гарантируется, сообщение может потеряться, и могут быть получены две копии одного и того же сообщения. Последний случай возникает, если для отправки сообщений в один адрес использовать два разных маршрута.

UDP не требует открывать соединение, и данные могут быть отправлены сразу же, как только они подготовлены. UDP не отправляет подтверждающие сообщения, поэтому данные могут быть получены или потеряны. Если при использовании UDP требуется надежная передача данных, ее следует реализовать в протоколе более высокого уровня.

**3.4 Разработка и реализация структуры приложения**

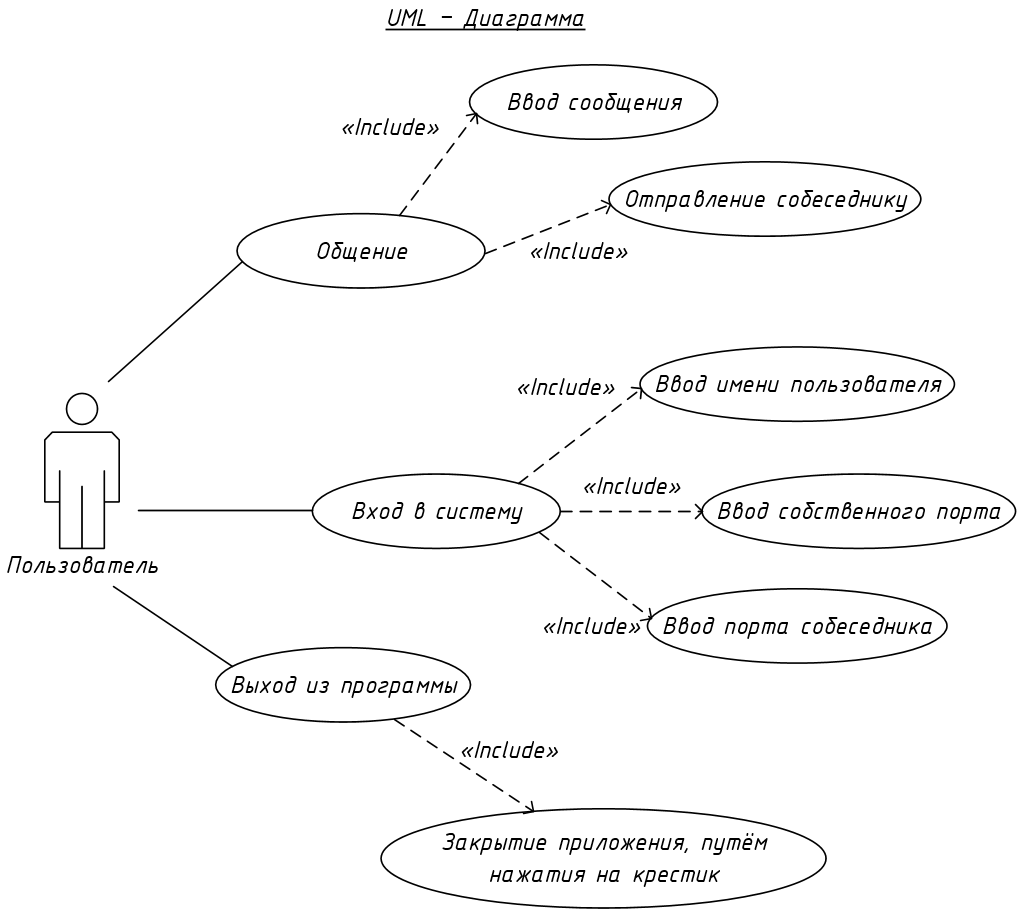


Рисунок 12 - Сценарий использования (Use case) программы

**3.5 Разработка и реализация алгоритмов приложения**

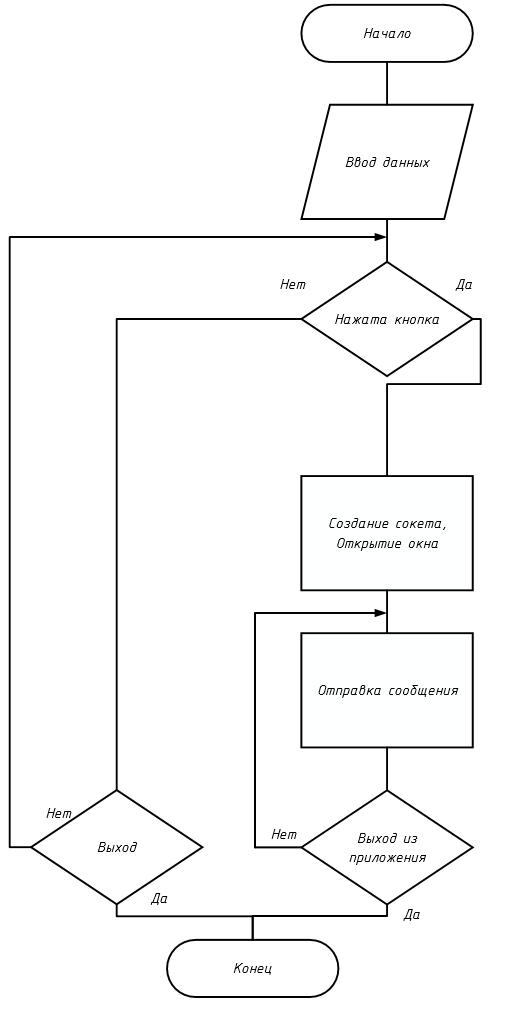


Рисунок 13 - Основная блок-схема, описывающая работу приложения в целом

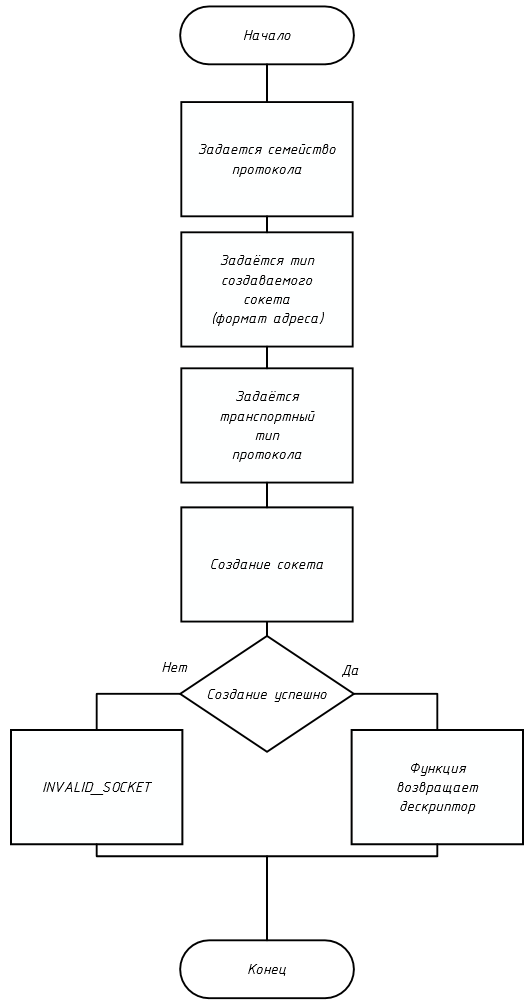


Рисунок 14 - Создание и проверка сокета

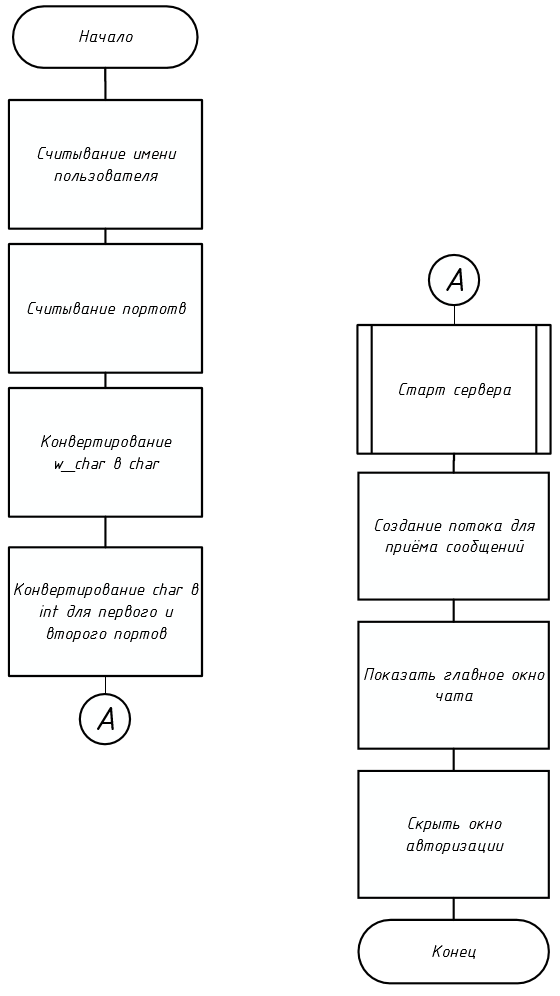


Рисунок 15 - Обработка кнопки «Войти»

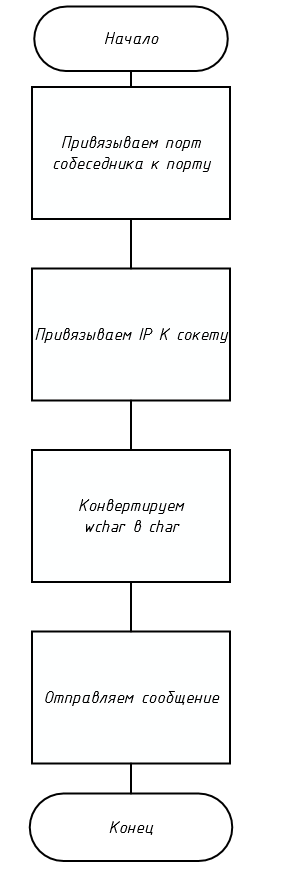


Рисунок 16 - Процесс отправления сообщения

## **3.5 Разработка интерфейса взаимодействия пользователя с системой**

Окно программы должно состоять из трёх текстовых полей для ввода информации: в первое поле необходимо записать имя клиента, во второе поле записать собственный порт, в третье же поле порт собеседника. Помимо этого, должна быть кнопка «Войти», при нажатии на которую открывается второе, новое окно – чат, в котором и происходит общение между двумя пользователями.

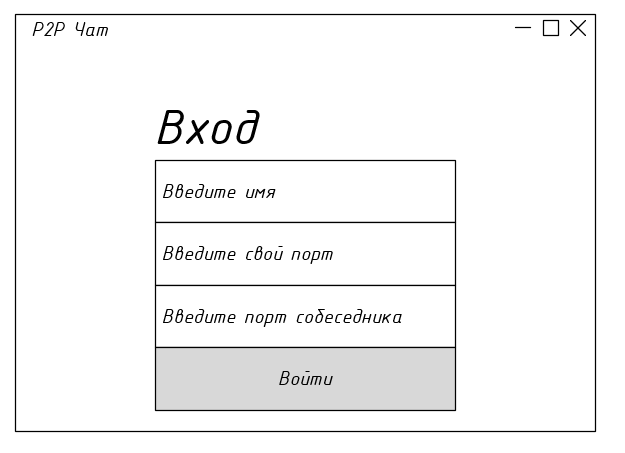


Рисунок 17 - Макет окна входа

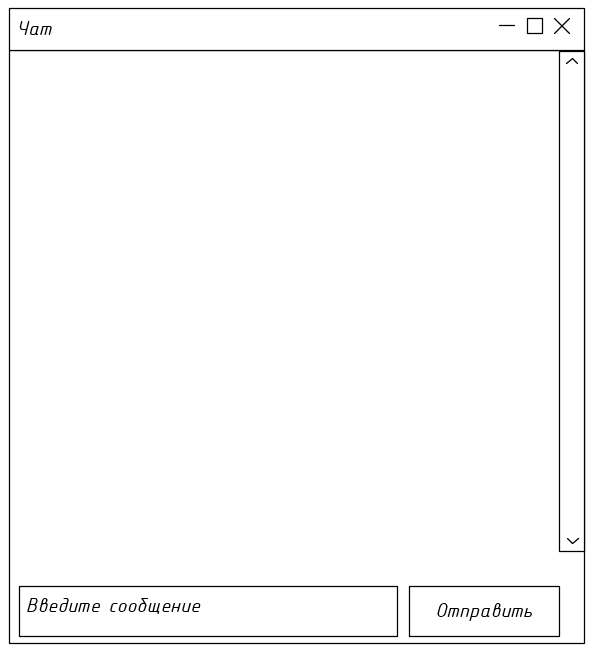


Рисунок 18 - Макет окна чата

# 

# 4. ПРОЕКТНО-ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

**4.1 Разработка руководства пользователя**

Данный программный продукт служит для мгновенного обмена сообщений между пользователями. Основными его преимуществами являются минимальный размер, нетребовательность в ресурсах.

Запуск программы осуществляется по открытию файла.exe , после запуска открывается окно приложения. Вводим всю необходимую информацию, чтобы войти в чат.

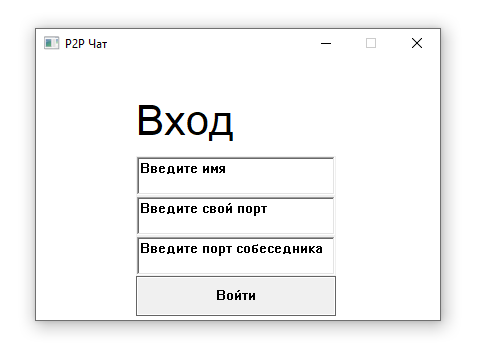


Рисунок 19 - Первоначальный вид окна

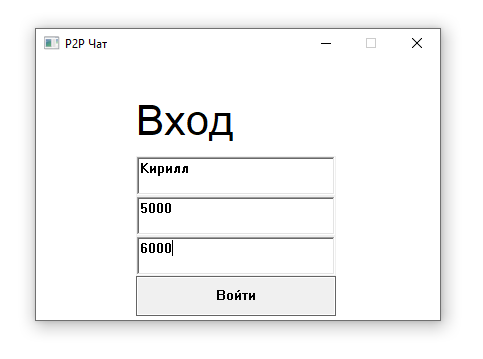


Рисунок 20 - Заполненное окно клиента

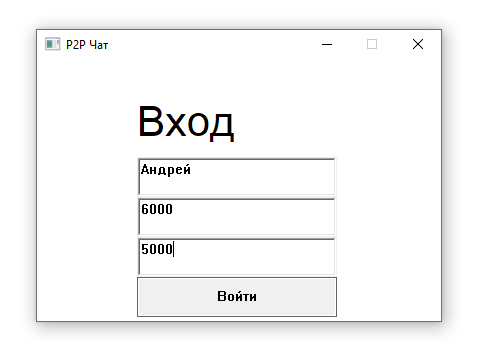


Рисунок 21 - Окно собеседника (второго клиента)

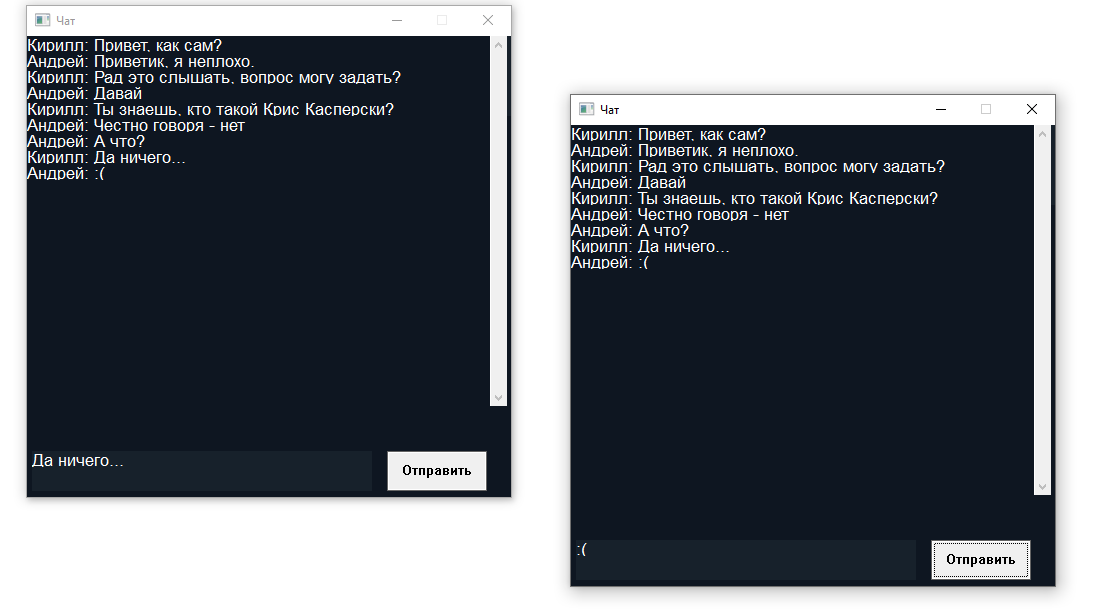


Рисунок 22 - Работающий чат

**4.2. Разработка руководства администратора**

Программа не требует никакой предварительной установки. Разархивируйте архивный файл, содержащий скомпилированные файлы, в удобное место.



Рисунок 23 - Заархивированное приложение

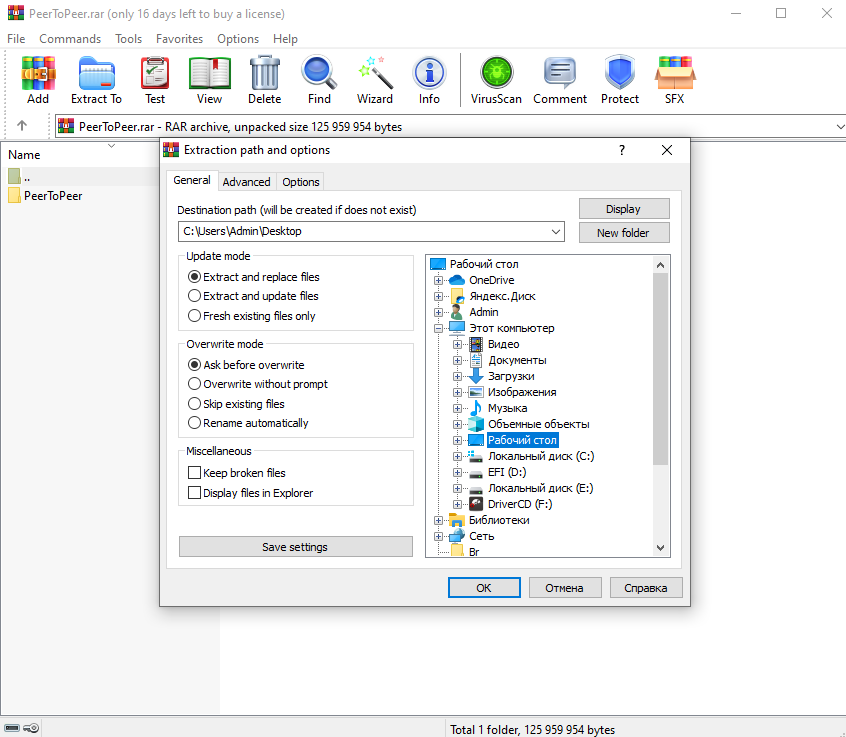


Рисунок 24 - Выбор места для разархивации проекта

Для того, чтобы запустить чат, вам необходимо запустить этот исполняемый файл .exe по пути \x64\Debug .

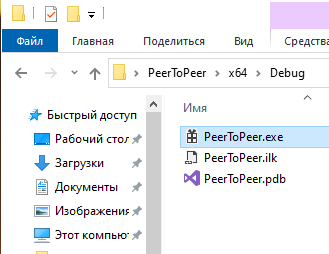


Рисунок 25 - Приложения находится в указанном месте

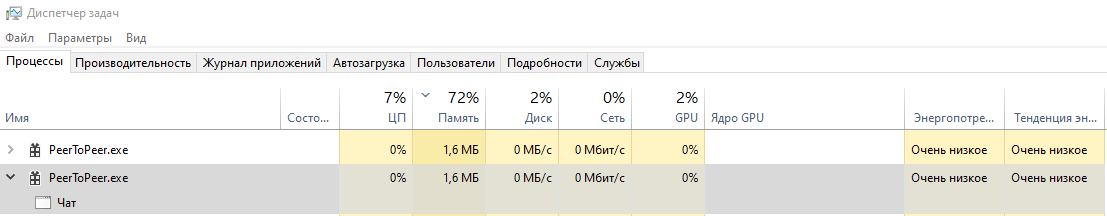
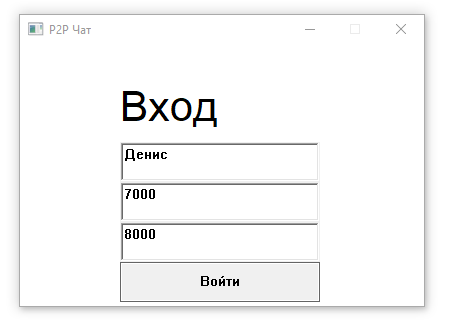


Рисунок 26 - Потребление ресурсов приложением

Использование клиента. После запуска Чат-клиента вы увидите небольшое окно. Оно необходимо для ввода псевдонима, от имени которого будут отправляться сообщения в дальнейшем, а также для установки соединения с собеседником. Ввод псевдонима является необходимым, чтобы не создавать путаницы между участниками чата. Ввод портов является обязательным, так как наличия неадресуемого пользователя в чате недопустимо. После этого вы увидите основной интерфейс программы – большое окно с набором определённых функций в нижней части. Новые сообщения будут появляться вверху. Для отправления сообщения наберите его в окне набора и нажмите кнопку «Отправить». После этого сообщение будет отправлено и отображено у каждого в окне чата.

# 



# Рисунок 27 – заполнение данных для входа

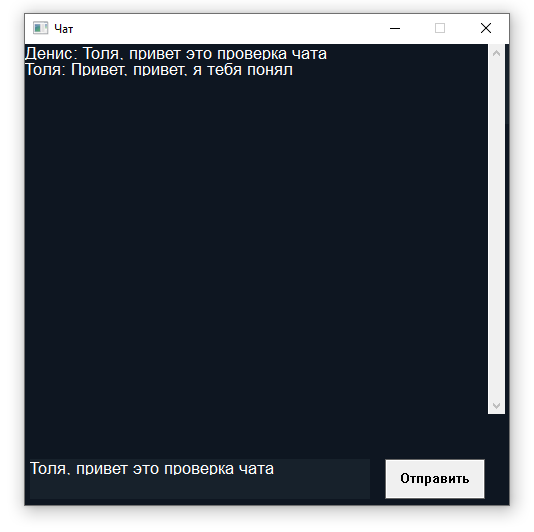


Рисунок 28 – отображение сообщений у клиента «Денис»

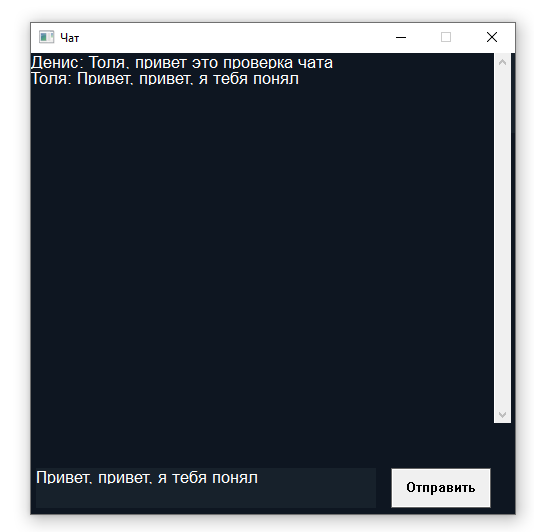


Рисунок 29 - Отображение сообщений у клиента «Толя»

Вход в приложение был выполнен успешно. Клиенты подключились друг к другу. Сообщения корректно отображаются в каждом чате. По завершении общения, приложение можно закрыть.

Если вы планируете пользоваться приложением на постоянной основе, то вам будет удобнее создать ярлык на рабочий стол, чтобы быстрее производить запуск приложения. Для этого необходимо нажать правой кнопкой мыши по исполняемому фалу, после этого выбрать пункт “Создать ярлык”. После этого можно перенести ярлык на рабочий стол, и при двойном нажатии на него, будет открываться приложение.

**4.3. Тестирование и отладка макета рабочей программы.**

Для работоспособности программы необходимо, чтобы каждый из пользователей правильно ввёл порт своего собеседника, иначе программа не будет работать корректно.

Каждый из клиентов допустили ошибки при заполнении порта собеседника.

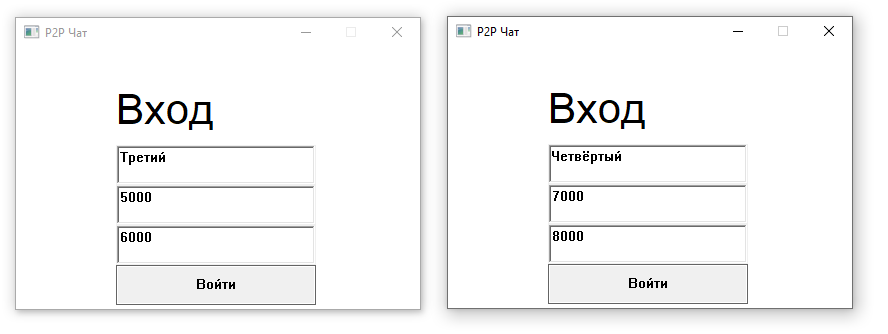


Рисунок 30 – Неправильное заполнение

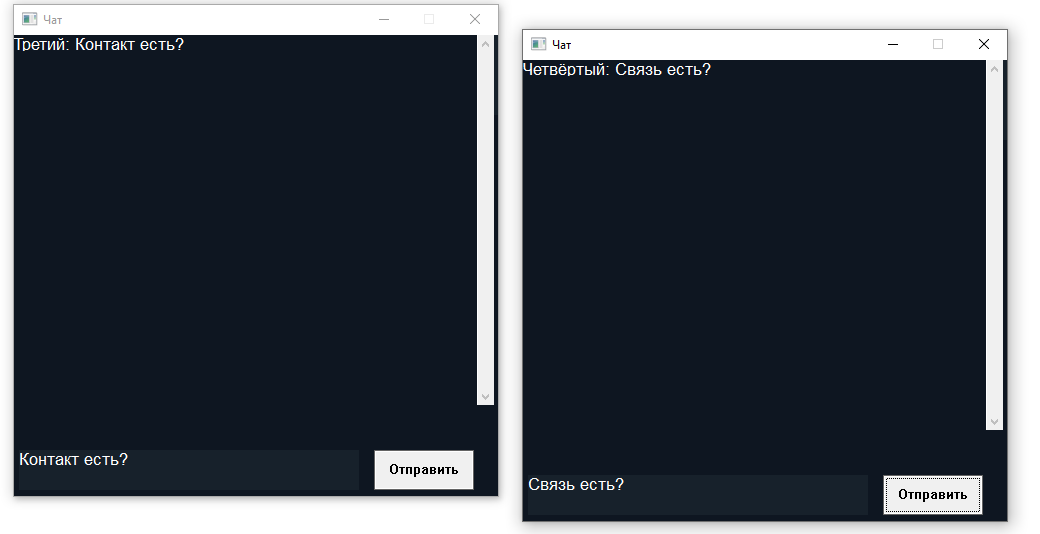


Рисунок 31 – Отсутствие связи между клиентами

# 5. ЗАКЛЮЧЕНИЕ

В ходе выполнения данной курсовой работы был разработана программа для обмена текстовыми сообщениями на основе WinAPI.

Работа выполнялась в несколько этапов: была разработана структура системы и реализовано прикладное оконное приложение на основе WinAPI.

Были сформированы навыки по разработке и реализации программного приложения с использованием интерфейса прикладного программирования (АРI) операционных систем.

Спроектированное приложение подойдет для задач, в ходе которых необходимо запустить файл в заданное пользователем время.

В будущем можно усовершенствовать разработанное приложение путем добавления дополнительного функционала (например, организовав передачу файлов от одного пользователя другому).

**6. СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТУРАТУРЫ**

1. Бречка Д.М. Операционные системы: в 3 ч. Ч. 1. Пакетные файлы и управление компьютером: учебно-методическое пособие. Омск: Омский государственный университет им. Ф.М. Достоевского, 2012 [https://e.lanbook.com/book/75382?category\_pk=1554#book\_name](https://e.lanbook.com/book/75382?category_pk=1554%23book_name)

2. Вирт Н., Гуткнехт Ю. Разработка операционной системы и компилятора. Проект Оберон. М.: ДМК Пресс, 2012 [https://e.lanbook.com/book/39992?category\_pk=1554#book\_name](https://e.lanbook.com/book/39992?category_pk=1554%23book_name)

3. Павловская Т.А. C/C++ . Процедурное и объектно-ориентированное программирование. Стандарт 3-го поколения. СПб.: Питер, 2015.

4. Платонов В.В. Программно-аппаратные средства защиты информации М.: Академия, 2014.

5. Сидоров В.Н., Сломинская Е.Н., Полникова Т.В., Макарова О.Ю. Оформление графической части выпускной квалификационной работы. Учебное пособие. М.: МГТУ им. Н.Э. Баумана, 2016.