Реферат

Пояснительная записка курсового проекта 33 с., 14 рис., 8 источников, 2 приложения.

ТЕХНОЛОГИЯ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ, РАЗРАБОТКА ПРИЛОЖЕНИЯ ТЕЛЕФОН, ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ, ДИАГРАММА ВАРИАНТОВ ИСПОЛЬЗОВАНИЯ, ДИАГРАММА КЛАССОВ, C#, WINDOWS FORMS

Объектом исследования являются программное обеспечение способное для телефона, которое позволяет производить вызов абонента с помощью терминала.

Целью курсового проекта является проектирование и разработка программного обеспечения «Телефон» с использованием UML диаграмм разного типа для полного представления реализуемого продукта.

К полученным результатам относится приложение WinForms, написанное на языке программирования C# в формате исполняемых файлов (.exe). Данное программное обеспечение было создано в среде разработки Microsoft Visual Studio.

При выполнении курсовой работы были выполнены все этапы создания программного продукта: от постановки задачи до практической реализации. В ходе выполнения курсовой работы были приобретены навыки работы со специализированной литературой, справочниками, стандартами

Содержание

Введение

Целью курсового проекта является проектирование и разработка программного продукта, который будет предоставлять возможность набора номера в терминал и выполнение звонка.

Задачи курсового проекта:

– изучить предметную область и провести анализ существующих решений;

– составить техническое задание проекта;

– спроектировать программный продукт;

– реализовать программный продукт;

– составить руководство пользователя.

В первой главе проекта указаны нормативные ссылки, которые были использованы при написании курсового проекта.

Во второй главе перечислены основные цель разработки, форма ее реализации, перечень потребностей конечного пользователя приложения.

В третьей главе содержится техническое задание проекта, на основе которого строится работа учебного комплекса.

В четвертой главе описан процесс проектирования программного продукта, выбор средств разработки.

В пятой главе описан процесс разработки программного продукта и руководство пользования готовым программным продуктом.

В заключении приводятся основные выводы из проведенного исследования, а также основные рекомендации по использованию ПО.

1 Нормативные ссылки

В данном курсовом проекте использованы следующие нормативные ссылки:

ГОСТ Р 1.5-2004 Стандартизация в Российской Федерации. Стандарты национальные Российской Федерации. Правила построения, изложения, оформления и обозначения.

ГОСТ Р 1.12-2004 Стандартизация в Российской Федерации. Термины и определения.

ГОСТ Р 7.0.5-2008 СИБИД. Библиографическая ссылка. Общие требования и правила составления.

ГОСТ Р ИСО 9000-2008 Системы менеджмента качества. Основные положения и словарь.

ГОСТ Р ИСО/МЭК 12207-99 Информационная технология. Процессы жизненного цикла программных средств.

ГОСТ 2.001-93 ЕСКД. Общие положения.

ГОСТ 19.103-77 ЕСПД. Обозначение программ и программных продуктов.

ГОСТ 19.105-78 ЕСПД. Обозначение программ и программных продуктов.

ГОСТ 19.401-78 ЕСПД. Текст программы. Требования к содержанию и оформлению.

ГОСТ 19.101-77 ЕСПД. Виды программ и программных документов.

2 Анализ предметной области

2.1 Цель приложения

Phone – программное обеспечение для набора номера абонента, состоящего из цифр, и выполнения вызова.

Ожидается, что программное обеспечение будет использоваться в качестве примера реализации программного обеспечения для телефона. А также в качестве примера разработанного приложения для студентов младших курсов, обучающихся на дисциплинах «Программирование», «Алгоритмы и структуры данных», «Объектно-ориентированное программирование», «Введение в программную инженерию», «Технологии разработки программного обеспечения».

2.2 Аналоги

Из аналогов данного приложения можно выделить приложение для персональных компьютеров Скайп (Skype).

Skype – бесплатное проприетарное программное обеспечение с закрытым кодом, обеспечивающее текстовую, голосовую и видеосвязь через Интернет между компьютерами (IP-телефония), опционально используя технологии пиринговых сетей, а также платные услуги для звонков на мобильные и стационарные телефоны. По состоянию на конец 2010 года у программы было 663 млн пользователей. Большинство разработчиков и 44 % работников общего отдела находятся в Эстонии – Таллине и Тарту.

Интерфейс приложения Скайп изображен на рисунке 2.2.1

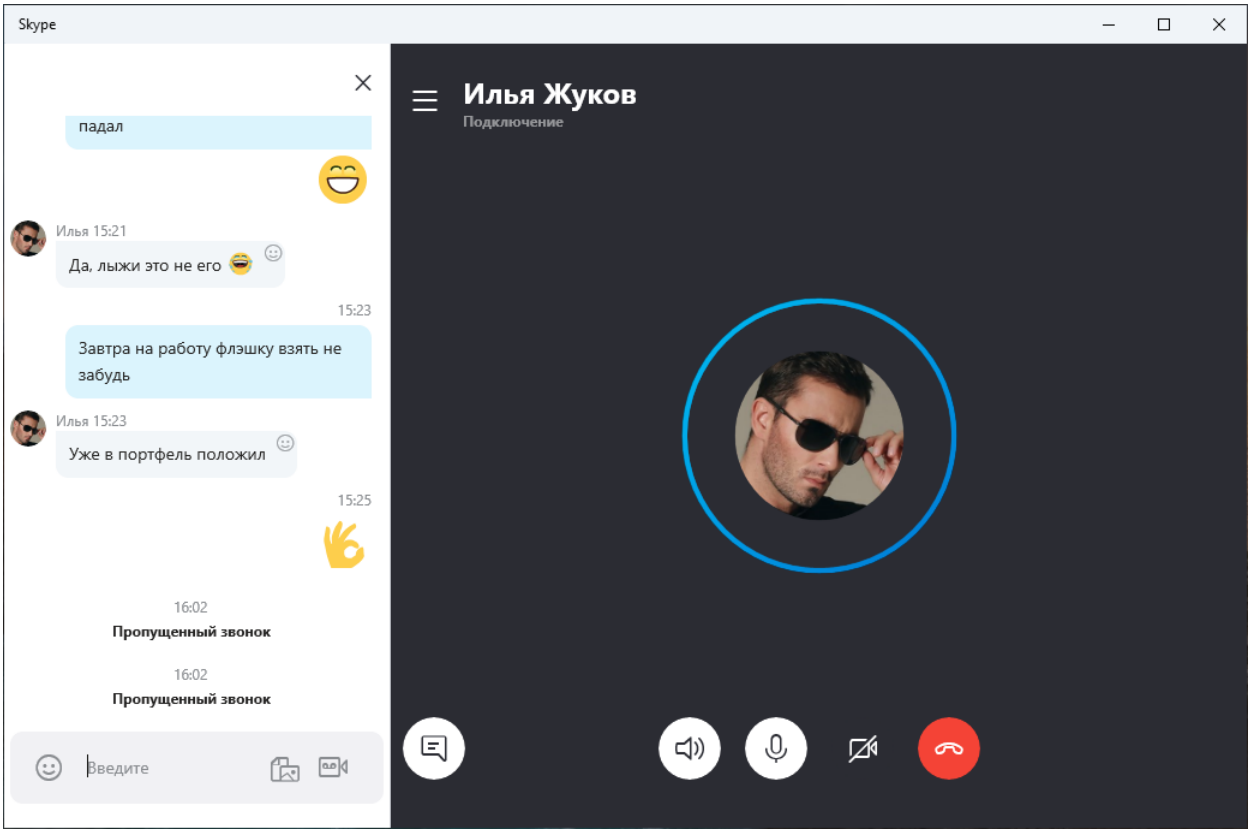

Рисунок 2.2.1 – Интерфейс приложения Скайп

3 Техническое задание

3.1 Введение

Настоящее техническое задание распространяется на разработку системы для осуществления звонков по набранному номеру телефона.

Разработанное программное обеспечение позволит пользователям с телефонами набирать цифровой номер абонентов и осуществлять вызов.

3.2 Основания для разработки

Указание о закреплении тем курсовых проектов №24-КТ от 23.09.2020.

3.3 Назначение разработки и область применения программы

Функциональным назначением программного продукта является предоставление пользователю возможности набора цифрового номера в терминале приложения и осуществления звонков по набранному номеру телефона.

Программа должна эксплуатироваться студентами и преподавателями в практическом и лабораторном курсе дисциплин кафедры ИСП. Программа позволит повысить уровень и качество учебного процесса студентов кафедры ИСП.

3.4 Требование к программе или программному изделию

Программа должна предоставлять пользователю следующий функционал:

- ввод номера телефона;
- осуществления звонка;

– осуществления отмены вызова.

3.5 Требования к надежности

– ПО должно иметь защиту от некорректных действий пользователей и ошибочных исходных данных;

– ПО не должно во время работы модифицировать свой код или код других программ;

– количество отказов ПО из-за не выявленных ошибок не должно превышать 1 отказа на 1000 сеансов работы с программой.

3.6 Время восстановления программы после отказа

– максимальное время устранения неисправностей в работе оборудования сервера системы – 1 час;

– максимальное время устранения неисправностей в работе по сервера системы – 30 минут;

– время устранения сбоев в работе системного и прикладного по рабочих станций – 12 часов.

3.7 Информационное обеспечение

– средства ввода данных в систему должны обеспечивать контроль правильности данных по типу;

– при модификации и удалении данных средства ведения ПО должны запрашивать подтверждение правильности выданных команд и временно сохранять предыдущую версию данных;

– при выполнении запросов, время которых превышает 10 секунд, на монитор пользователя должен выдаваться транспарант с извещением о вынужденном ожидании.

3.9 Требования к составу и параметрам технических средств

В состав технических средств должны входить IBM-совместимый персональный компьютер, выполняющий роль сервера, включающий в себя:

– 32-разрядный или 64-разрядный процессор с тактовой частотой 1 ГГц или выше;

– 1 ГБ ОЗУ для 32-разрядного процессора или 2 ГБ ОЗУ для 64-разрядного процессора;

– 16 ГБ для 32-рязрядной системы или 20 ГБ для 64-разрядной системы свободного места на жестком диске;

– операционную систему Windows 7 или новее.

3.10 Предварительный состав программной документации

Состав программной документации должен включать в себя:

– техническое задание;

– программный продукт и методику испытаний;

– руководство оператора.

3.11 Экономические преимущества разработки

Ориентировочная экономическая эффективность не рассчитывается в виду уникальности предъявляемых требований к разработке.

3.12 Стадии разработки

Разработка должна быть проведена в 3 стадии:

– разработка технического задания;

– рабочее проектирование;

– внедрение.

3.13 Этапы разработки

На стадии разработки ТЗ должен быть выполнен этап разработки, согласования и утверждение технического задания. На стадии рабочего проектирования должны быть выполнены следующие этапы работ:

– разработка программы;
– разработка программной документации;
– испытания программы.

4 Проектирование программного обеспечения

4.1 UML диаграммы

Диаграмма в языке моделирования UML – наглядное представление некоей совокупности элементов модели системы в виде графа, на котором дуги (отношения) связывают вершины (сущности). В своём графическом виде различные виды диаграмм UML (диаграммы классов, компонентов, объектов и др.) применяются для визуализации разных аспектов устройства или поведения моделируемой системы.

Диаграмма вариантов использования (use case) представляет собой последовательность действий, выполняемых системой в ответ на событие, инициализируемое некоторым внешним объектом. Вариант использования описывает типичное взаимодействие между пользователем и системой. В простейшем случае вариант использования определяется в процессе обсуждения с пользователем тех функций, которые он хотел бы реализовать.

Диаграмма вариантов использования данного проекта изображена на рисунке 4.1.1.

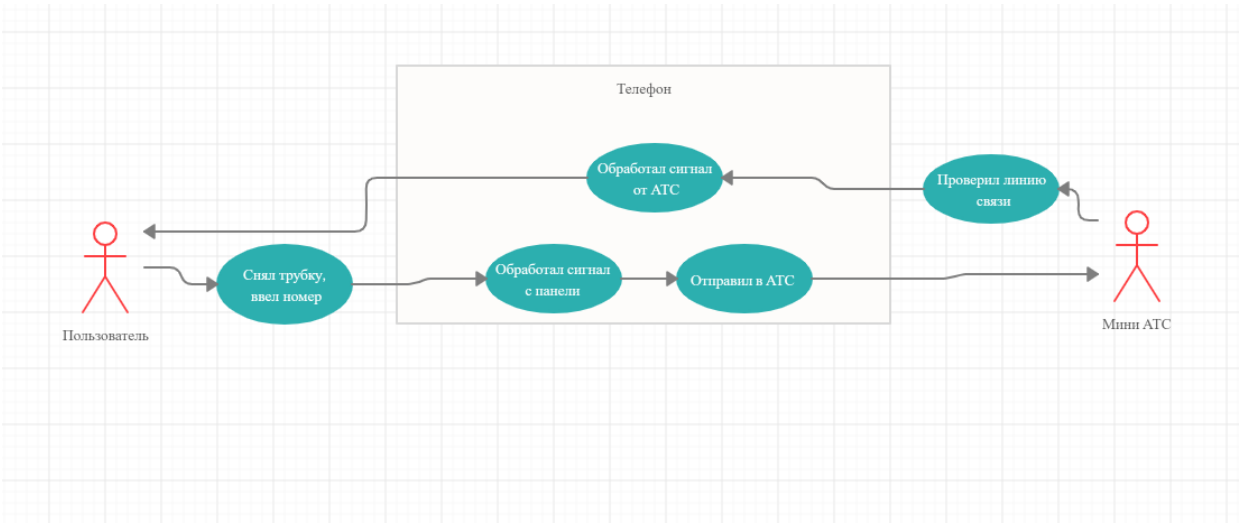

Рисунок 4.1.1 – Диаграмма вариантов использования

Диаграмма классов – это центральная методика моделирования, которая используется практически во всех объектно-ориентированных

методах. Эта диаграмма описывает типы объектов в системе и различные виды статических отношений, которые существуют между ними.

Три наиболее важных типа отношений в диаграммах классов (на самом деле их больше), это:

1. Ассоциация, которая представляет отношения между экземплярами типов, к примеру, человек работает на компанию, у компании есть несколько офисов.

2. Наследование, которое имеет непосредственное соответствие наследованию в Объектно-Ориентированном дизайне.

3. Агрегация, которая представляет из себя форму композиции объектов в объектно-ориентированном дизайне.

Диаграмма классов данного проекта изображена на рисунке 4.1.2.

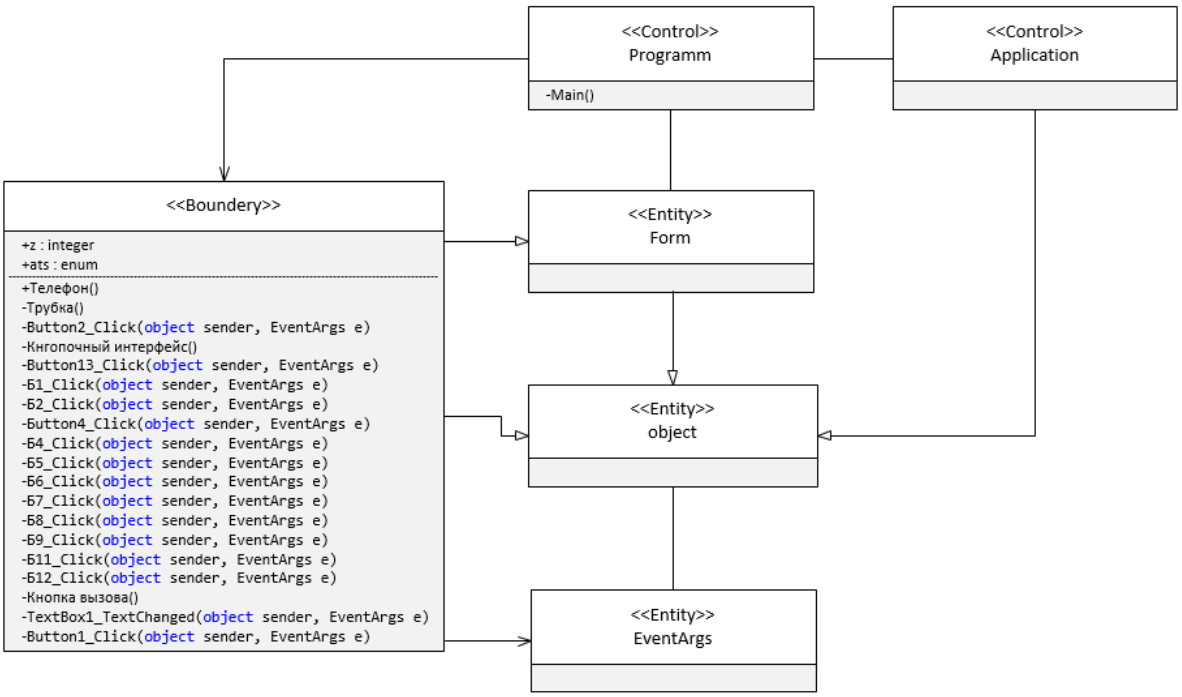

Рисунок 4.1.2 – Диаграмма классов

Диаграмма пакетов – это структурная схема UML, которая показывает пакеты и зависимости между ними. Она позволяет отображать различные виды системы, например, легко смоделировать многоуровневое приложение.

Диаграмма пакетов данного проекта изображена на рисунке 4.1.3.

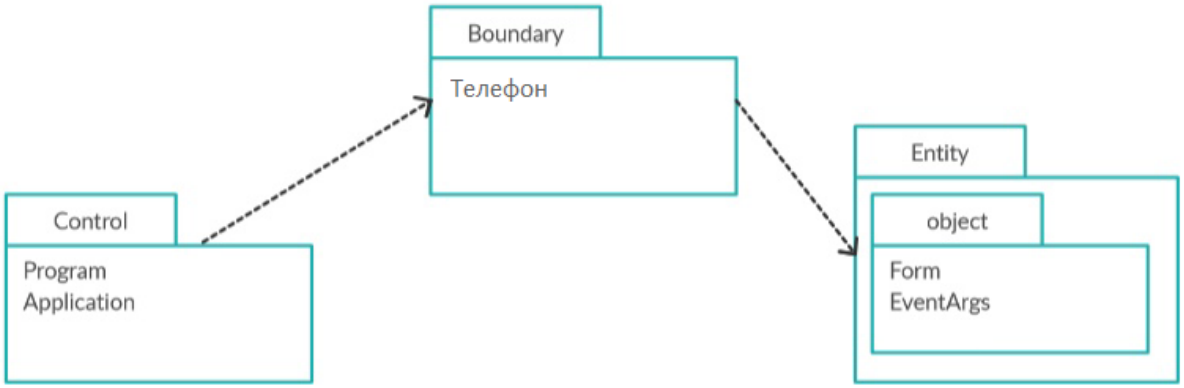

Рисунок 4.1.3 – Диаграмма пакетов

Диаграмма состояний – это тип диаграммы, используемый в UML для описания поведения систем, который основан на концепции диаграмм состояний Дэвида Харела. Диаграммы состояний отображают разрешенные состояния и переходы, а также события, которые влияют на эти переходы. Она помогает визуализировать весь жизненный цикл объектов и, таким образом, помогает лучше понять системы, основанные на состоянии.

Диаграмма состояний данного проекта изображена на рисунке 4.1.4.

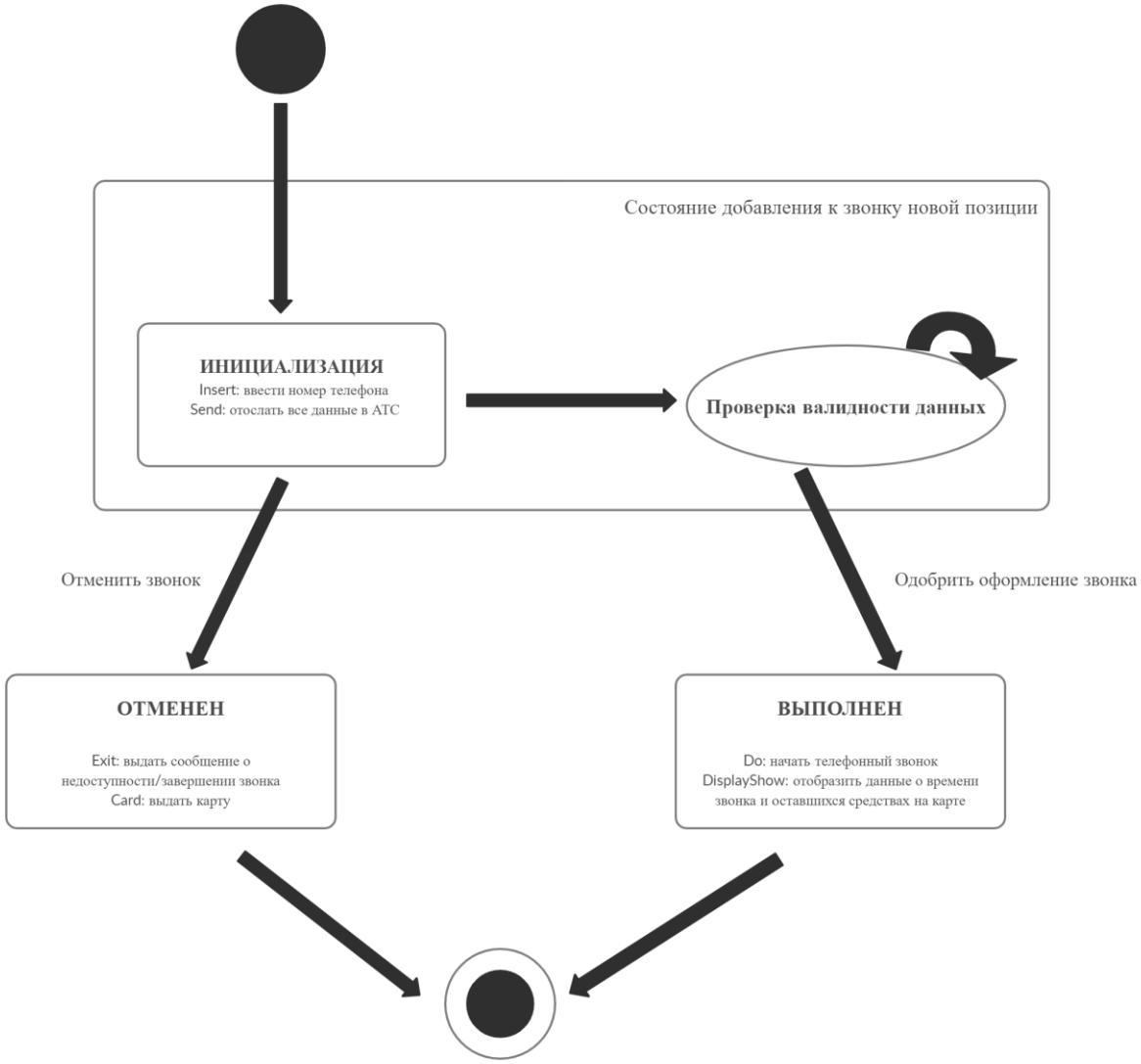

Рисунок 4.1.4 – Диаграмма состояний

На языке унифицированного моделирования диаграмма компонентов показывает, как компоненты соединяются вместе для формирования более крупных компонентов или программных систем.

Она иллюстрирует архитектуры компонентов программного обеспечения и зависимости между ними.

Эти программные компоненты включают в себя компоненты времени выполнения, исполняемые компоненты, а также компоненты исходного кода.

Диаграмма компонентов данного проекта изображена на рисунке 4.1.5.

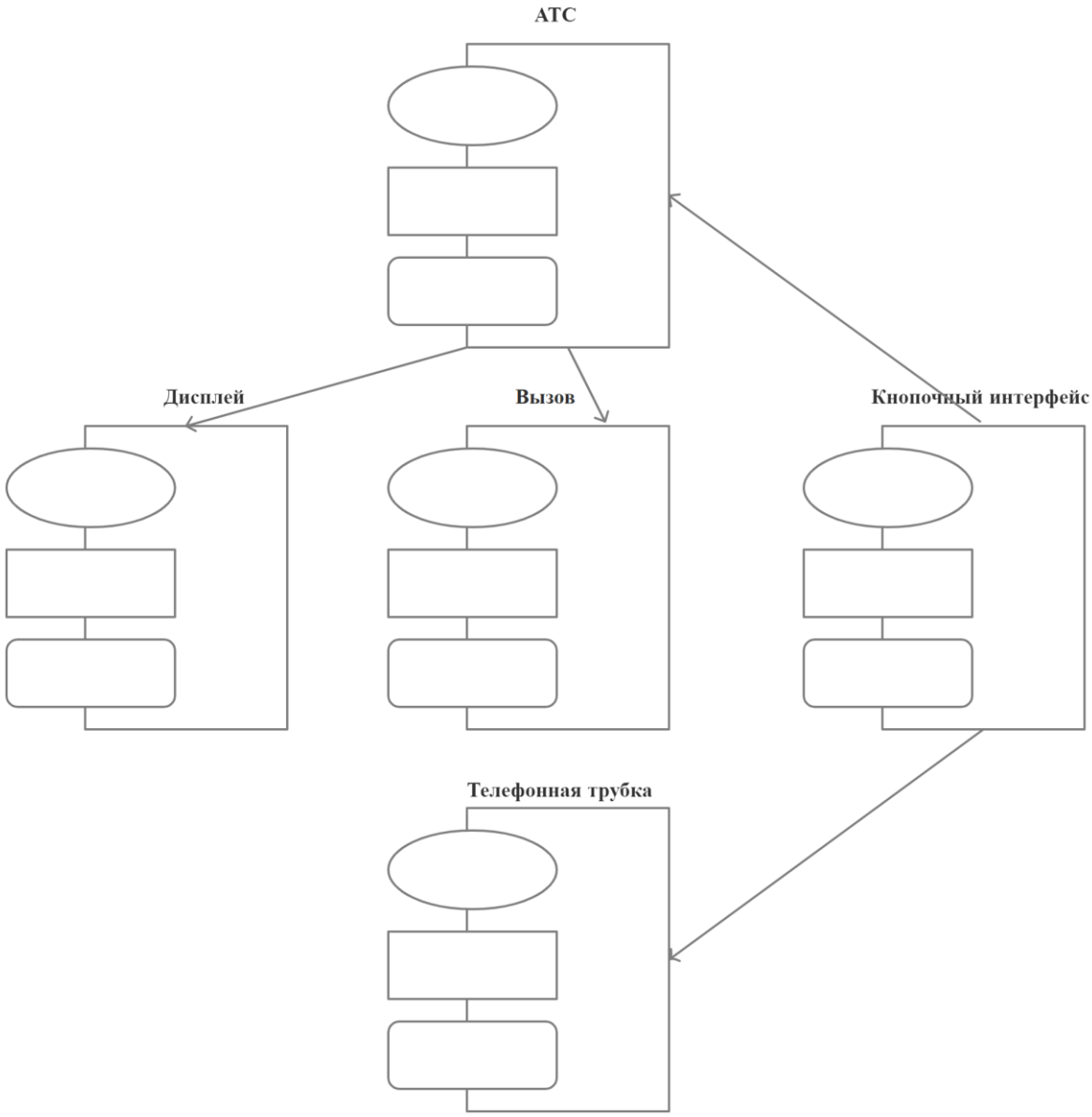

Рисунок 4.1.5 – Диаграмма компонентов

4.2 Проектирование интерфейса приложения

Одним из важных этапов разработки – это проектирование интерфейса программы. Это то, как будет выглядеть конечный продукт для пользователя, но без детализации, просто в виде схематичных блоков. Данный макет может подсказать нам, какие элементы управления необходимо разработать, какие контейнеры необходимы для размещения

всей информации. Макет интерфейса приложения изображен на рисунке 4.2.1 и 4.2.2. Проектирование основного интерфейса приложения было реализовано при помощи ресурса [2].

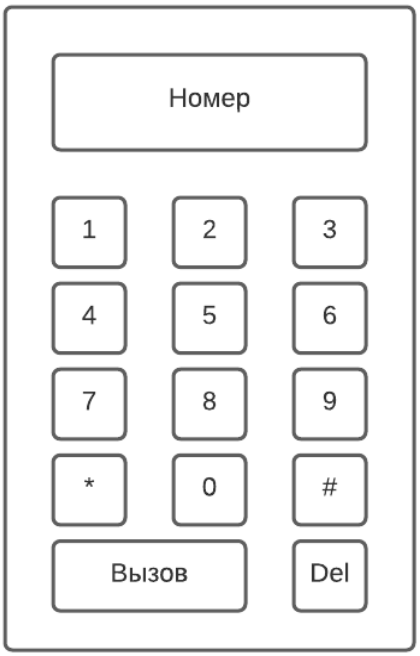

Рисунок 4.2.1 – Макет интерфейса окна приложения набора номера

Рисунок 4.2.2 – Макет интерфейса окна приложения вызова

4.3 Выбор языка программирования

После того, как мы увидели то, как схематически должен выглядеть готовый продукт для пользователя, нужно определиться при помощи каких средств программирования этого можно достичь. Для написания данного проекта будет использоваться язык программирования C#, библиотеки для работы с средствами построения графического интерфейса Windows Forms и среда разработки Microsoft Visual Studio 2019 для удобного построения графического дизайна и упрощения написания кода.

5 Реализация программного продукта

5.1 Описание методов программирования

В приложении существует две основные сцены:

Первая сцена включает в себя панель с набором кнопок для набора номера телефона и кнопку для вызова.

Вторая сцена включает в себя панель, на которой присутствует информация о текущем вызове, такая как номер, страна и состояние вызова, а также кнопка сброса вызова.

Для реализации этих сцен использовались два элемента Panel, которые меняли свое свойство Visible в зависимости от того, какую сцену необходимо было показать пользователю.

Для визуально красивого представления обеих сцен телефона использовалась картинка уже существующей панели набора номер и вызова. Для того чтобы установить данную картинку на задний фон сцены использовалось свойство BackgroundImage.

Информация по работе с элементом управления Panel найдена на электронных ресурсах [3] и [4].

Для реализации ввода номера телефона использовались элементы управления Button, данный элемент представляет из себя кнопку, у которой есть событие Click. К этому событию у каждой кнопки был привязан вызов функции обработчика. Которая вызывала метод phoneTextAdd(параметр) с параметром равным значению данной кнопки – это весь диапазон цифр.

Код метода phoneTextAdd:

```
private void phoneTextAdd(char digit)
{
  if (PhoneNumber.Length >= 11) return;
  PhoneNumber += digit;
}
```

Данный метод добавлял к свойству PhoneNumber новую цифру при условии, что количество цифр в нем не превышало 11.

У свойства PhoneNumber присутствует обработчик изменения значений, который не просто сохраняет новое значение набранного номера, но и выполняет красивый вывод номера на экран в формате: X (XXX) XXX-XX-XX. Код данного обработчика:

```
private string PhoneNumber
{
  get { return _phoneNumber; }
  set
  {
    _phoneNumber = value;

    string str = "";
    if (_phoneNumber.Length > 0) str = _phoneNumber[0] + " ";

    if (_phoneNumber.Length > 3) str += string.Format("({0}) ",
_phoneNumber.Substring(1, 3));
    else if (_phoneNumber.Length > 1) str +=
string.Format("({0}) ", _phoneNumber.Substring(1));

    if (_phoneNumber.Length > 6) str +=
_phoneNumber.Substring(4, 3);
    else if (_phoneNumber.Length > 4)  str +=
_phoneNumber.Substring(4);

    if (_phoneNumber.Length > 8) str += "-" +
_phoneNumber.Substring(7, 2);
    else if (_phoneNumber.Length > 7)  str += "-" +
_phoneNumber.Substring(7);

    if (_phoneNumber.Length > 9) str += "-" +
_phoneNumber.Substring(9);

    numberDialing.Text = str;
  }
}
```

Информация по работе с элементами управления Button найдена на электронном ресурсе [5], информация по работе со свойствами классов найдена на электронном ресурсе [6].

Для вывода номера телефона на экран использовался элемент управления Label и его свойство Text, в который и помещался уже отформатированный номер телефона. Информация по работе с элементом управления Label найдена на электронном ресурсе [7].

Для отслеживания времени продолжительности звонка, его остановки и динамического отображения вызова использовался визуально скрытый элемент управления Timer. У данного элемента было установлено свойство Interval равное 500 миллисекунд, это означает что каждые 500 миллисекунд будет происходить один тик таймера. Для каждого такого тика происходит вызов метода, обработчика события его изменения. Код данного метода:

```
private byte ticks = 0;
private void timer1_Tick(object sender, EventArgs e)
{
  ticks++;
  if (ticks < 30)
  {
    string str = "Набор номера";
    sbyte dots = (sbyte)(ticks & 0b11);
    while (dots-- > 0) str += ".";
    status.Text = str;
  }
  else if (ticks < 40)
  {
    status.Text = "Не удалось дозвониться";
  }
  else
  {
    timer1.Stop();
    calling.Visible = false;
    ticks = 0;
  }
}
```

Информация по работе с элементов управления Timer найдена на электронном ресурсе [8].

Весь код программного продукта представлен в приложении А.

5.2 Руководство пользователя

Как только пользователь запустит приложение «Phone» перед ним откроется главное окно программы (см. пример на рисунке 5.2.1).

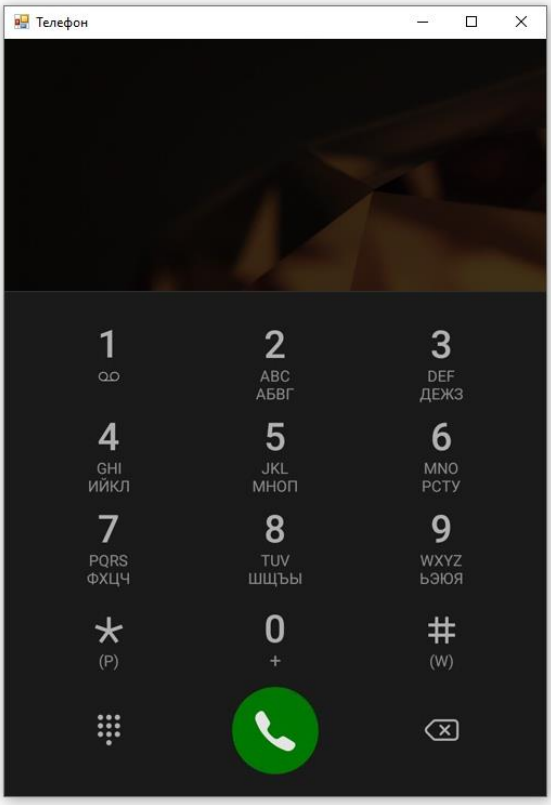

Рисунок 5.2.1 – Главное окно приложения

По началу на главном окне не будет ничего кроме цифровой клавиатуры. После того как пользователь начнет вводить номер он начнет отображаться на панели над цифровой клавиатурой (см. пример на рисунке 5.2.2).

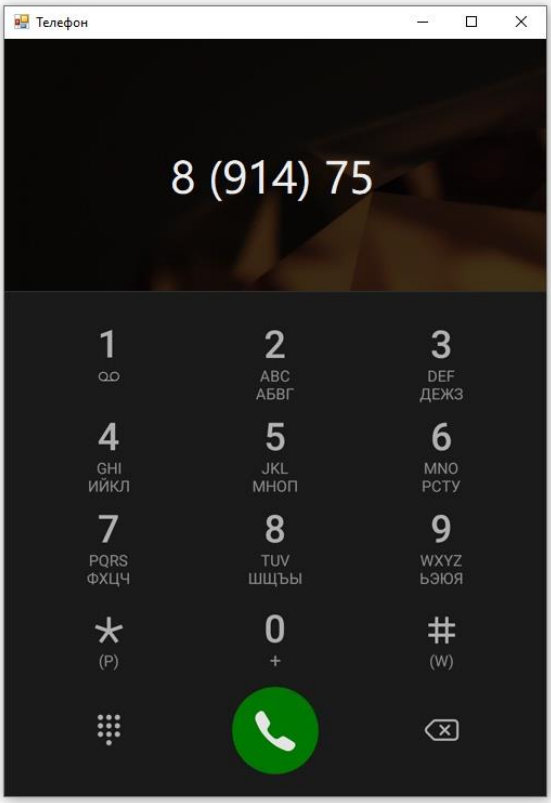

Рисунок 5.2.2 – Главное окно с частью набранного номера

Для того чтобы осуществить вызов по набранному номеру, необходимо нажать на зеленую кнопку «Вызов», после чего интерфейс приложения изменится как показано на рисунке 5.2.3.

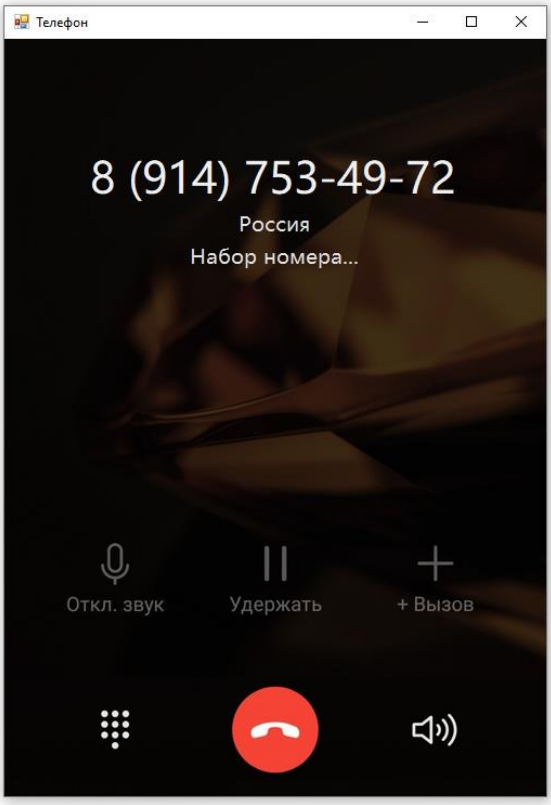

Рисунок 5.2.3 – Интерфейс вызова абонента

Для того чтобы сбросить вызов необходимо нажать на красную кнопку снизу, после чего вы перейдете на панель с вводом номера телефона как показано на рисунке 5.2.4.

Нажимать на данную кнопку не обязательно, вызова абонента закончиться автоматически через определенное время с уведомлением «Не удалось дозвониться», как это показано на рисунке 5.2.5.

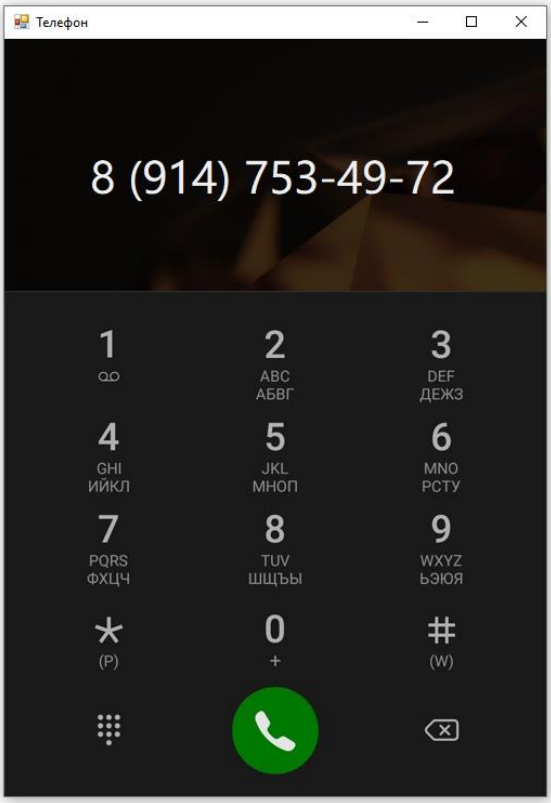

Рисунок 5.2.4

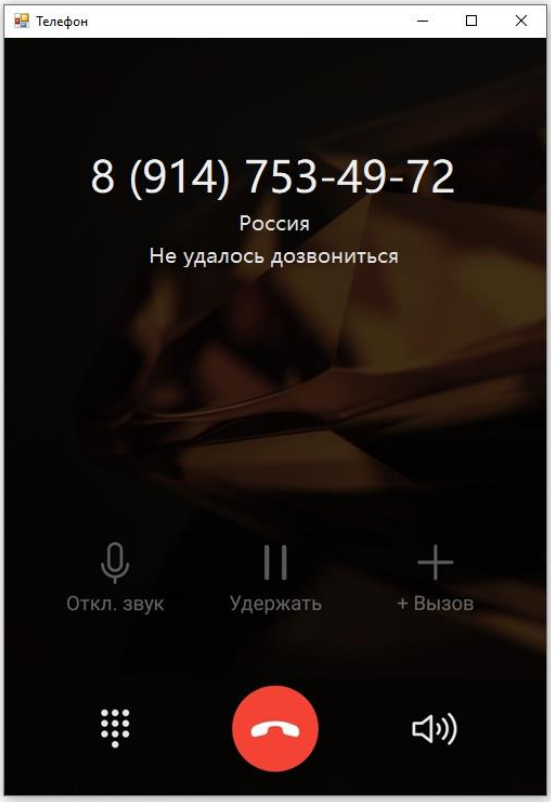

Рисунок 5.2.5 – Пример автоматического завершения вызова

Заключение

В результате курсового проекта по дисциплине «Технологии разработки программного обеспечения» разработан программное обеспечение, позволяющее осуществлять набор номера телефона и выполнять вызов.

Цель курсового проекта, которая заключается в проектировании и разработке программного продукта, выполнена.

В ходе работы были выполнены следующие задачи:

- произведено изучение предметной области;
- произведен анализ существующих решений и аналогов;
- техническое задание проекта составлено;
- программный продукт спроектирован;
- программный продукт разработан;
- составлено руководство пользователя и пояснительная записка проекта.

В ходе разработки использовались такие программные средства, как язык программирования C#, библиотека для работы с графическим интерфейсом управления Windows Forms и среда, которая все это в себе совмещает и предоставляет быстрый и простой доступ ко всем средствам тестирования и разработки программного кода Microsoft Visual Studio.

Список используемых источников

1. Техническое задание. Требование к содержанию и оформлению. ГОСТ 19.201-78 [Текстовый документ]
2. Lucid [Электронный ресурс] URL: https://lucid.app/ (дата обращения 26.02.21)
3. Panel Класс [Электронный ресурс] URL: https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.forms.panel?view=netframework-4.8 (дата обращения 27.02.21)
4. Элементы GroupBox, Panel и FlowLayoutPanel [Электронный ресурс] URL: https://metanit.com/sharp/windowsforms/3.2.php (дата обращения 27.02.21)
5. Button Класс [Электронный ресурс] URL: https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.forms.button?view=netframework-4.8 (дата обращения 06.03.21)
6. Свойства [Электронный ресурс] URL: https://metanit.com/sharp/tutorial/3.4.php (дата обращения 06.03.21)
7. Label Класс [Электронный ресурс] URL: https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.forms.label?view=netframework-4.8 (дата обращения 06.03.21)
8. Timer Класс [Электронный ресурс] URL: https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.forms.timer?view=netframework-4.8 (дата обращения 08.03.21)

Приложение А

Листинг

Файл Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Coursework_Phone
{
  static class Program
  {
    /// <summary>
    /// Главная точка входа для приложения.
    /// </summary>
    [STAThread]
    static void Main()
    {
      Application.EnableVisualStyles();
      Application.SetCompatibleTextRenderingDefault(false);
      Application.Run(new Form1());
    }
  }
}
```

Файл Form1.cs

```
using System;
using System.Drawing;
using System.Windows.Forms;

namespace Coursework_Phone
{
  public partial class Form1 : Form
  {

    private string _phoneNumber;
    private string PhoneNumber
    {
      get { return _phoneNumber; }
      set
      {
        _phoneNumber = value;

        string str = "";
        if (_phoneNumber.Length > 0) str = _phoneNumber[0] + " ";
```

```
        if (_phoneNumber.Length > 3) str += string.Format("({0}) ",
_phoneNumber.Substring(1, 3));
        else if (_phoneNumber.Length > 1) str += string.Format("({0})
", _phoneNumber.Substring(1));

        if (_phoneNumber.Length > 6) str += _phoneNumber.Substring(4,
3);
        else if (_phoneNumber.Length > 4)  str +=
_phoneNumber.Substring(4);

        if (_phoneNumber.Length > 8) str += "-" +
_phoneNumber.Substring(7, 2);
        else if (_phoneNumber.Length > 7)  str += "-" +
_phoneNumber.Substring(7);

        if (_phoneNumber.Length > 9) str += "-" +
_phoneNumber.Substring(9);

        numberDialing.Text = str;
      }
    }

    public Form1()
    {
      InitializeComponent();
      _phoneNumber = numberDialing.Text;

      for (int i = 0; i < dialing.Controls.Count; i++)
        SetEllipseRegion(dialing.Controls[i]);
      for (int i = 0; i < calling.Controls.Count; i++)
        SetEllipseRegion(calling.Controls[i]);
    }
    private void SetEllipseRegion(Control control)
    {
      System.Drawing.Drawing2D.GraphicsPath gPath = new
System.Drawing.Drawing2D.GraphicsPath();
      gPath.AddEllipse(0, 0, control.Width, control.Height);
      control.Region = new Region(gPath);
    }
    private void buttonDrop_Click(object sender, EventArgs e)
    {
      calling.Visible = false;
    }

    private void buttonCall_Click(object sender, EventArgs e)
    {
      status.Text = "Набор номера";
      numberCalling.Text = numberDialing.Text;
      calling.Visible = true;
      timer1.Start();
    }
```

```
    private byte ticks = 0;
    private void timer1_Tick(object sender, EventArgs e)
    {
      ticks++;
      if (ticks < 30)
      {
        string str = "Набор номера";
        sbyte dots = (sbyte)(ticks & 0b11);
        while (dots-- > 0) str += ".";
        status.Text = str;
      }
      else if (ticks < 40)
      {
        status.Text = "Не удалось дозвониться";
      }
      else
      {
        timer1.Stop();
        calling.Visible = false;
        ticks = 0;
      }
    }



    private void phoneTextAdd(char digit)
    {
      if (PhoneNumber.Length >= 11) return;
      PhoneNumber += digit;
    }

    private void button1_Click(object sender, EventArgs e) {
phoneTextAdd('1'); }

    private void button2_Click(object sender, EventArgs e) {
phoneTextAdd('2'); }

    private void button3_Click(object sender, EventArgs e) {
phoneTextAdd('3'); }

    private void button4_Click(object sender, EventArgs e) {
phoneTextAdd('4'); }

    private void button5_Click(object sender, EventArgs e) {
phoneTextAdd('5'); }

    private void button6_Click(object sender, EventArgs e) {
phoneTextAdd('6'); }
```

```
    private void button7_Click(object sender, EventArgs e) {
phoneTextAdd('7'); }

    private void button8_Click(object sender, EventArgs e) {
phoneTextAdd('8'); }

    private void button9_Click(object sender, EventArgs e) {
phoneTextAdd('9'); }

    private void button0_Click(object sender, EventArgs e) {
phoneTextAdd('0'); }

    private void buttonDelete_Click(object sender, EventArgs e)
    {
      if (numberDialing.Text.Length > 0)
        PhoneNumber = PhoneNumber.Remove(PhoneNumber.Length-1);
    }
  }
}
```

Приложение Б