УДК 004.457

ИСПОЛЬЗОВАНИЕ ТЕХНОЛОГИИ .NET MAUI и ML.NET ДЛЯ АВТОМАТИЗАЦИИ ПРОЦЕССА УЧЕТА РЕМОНТА И ОБСЛУЖИВАНИЯ ОБОРУДОВАНИЯ И ПО ИТ-ИНФРАСТРУКТУРЫ

Кузменков А.О., студент гр. ИТб-201, IV курс
Ванеев О.Н., доцент (к.н.) кафедры ИиАПС
Кузбасский государственный технический университет
имени Т. Ф. Горбачева
г. Кемерово

В современном мире люди активно пользуются разными устройствами от персональных компьютеров и ноутбуков до смартфонов и айфонов. При этом возникает проблема интеграции приложений под разные платформы устройств. Существует вариант с общей базой данной и разными клиентами под каждую платформу, однако это довольно трудозатратное дело. Для решения этой проблемы можно использовать фреймворки, объединяющие API-интерфейсы разных платформ. Одним из таких является .NET MAUI.

.NET MAUI – кроссплатформенная платформа для создания мобильных и классических приложений с использования языка разметки пользовательского интерфейса XAML и языка программирования C#. Краткий принцип работы технологии представлен на рисунке 1.

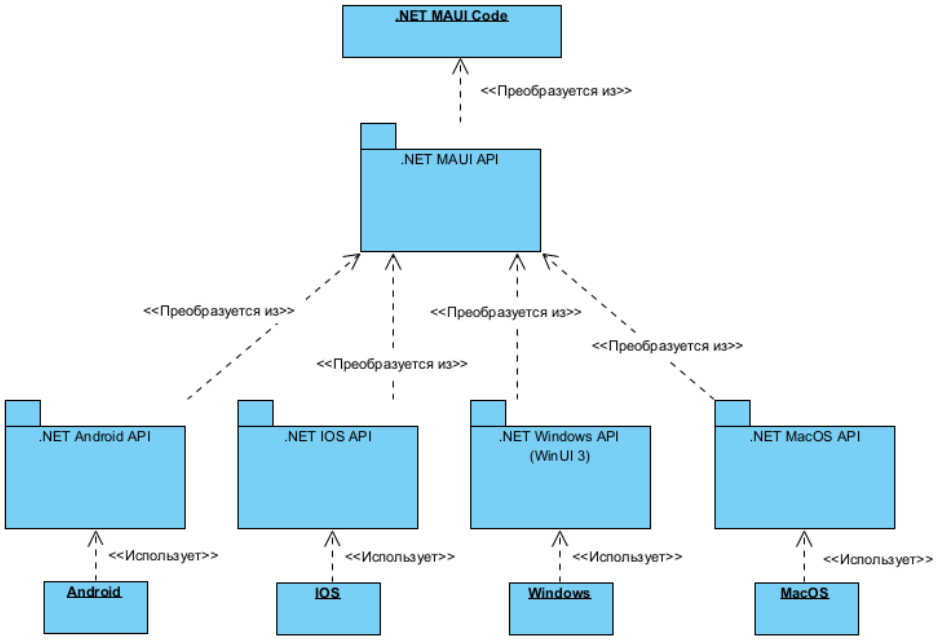

Рисунок 1 – Диаграмма UML упрощенного принципа работы .NET MAUI.

Рассмотрим функции автоматизируемого бизнес-процесса:

- Сбор информации о состоянии информационных систем и оборудования ИТ-инфраструктуры;

- Качественное реагирование на возникающие проблемы с использованием информационных систем предприятия и оборудования ИТ-инфраструктуры;
- Ремонт или замена вышедшего из строя оборудования ИТ-инфраструктуры;
- Формирование отчетов деятельности процесса;
- Анализ выходных данных для своевременного превентивного обслуживания и профилактических действий проблемного оборудования.

Входная информация – заявки с описанием проблемы программного обеспечения или оборудования, планы на обслуживание ИТ-инфраструктуры.

IDEF0-диаграмма процесса представлена на рисунке 2.

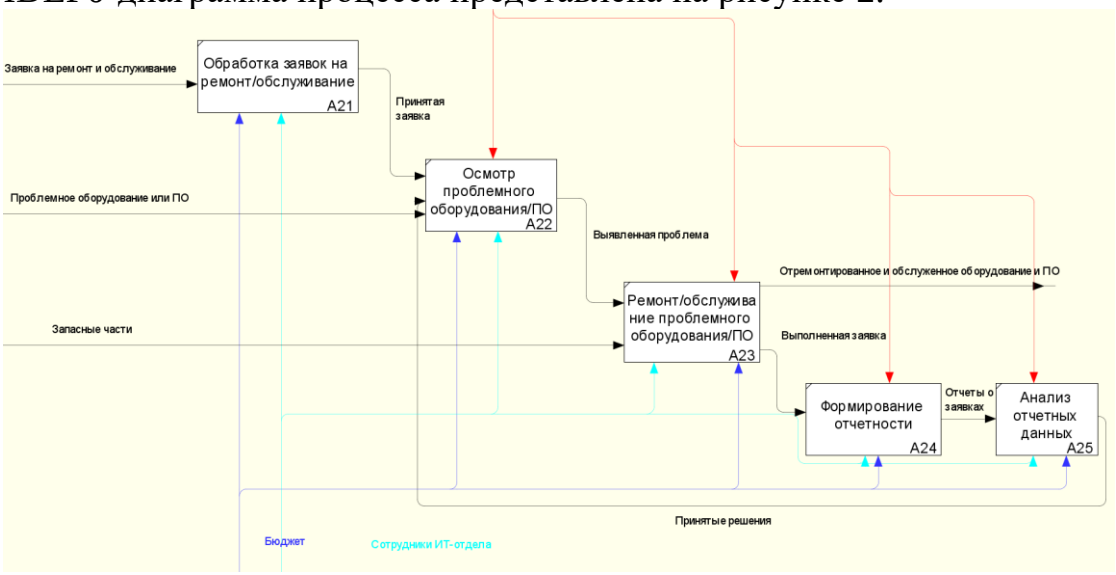

Рисунок 2 – Диаграмма IDEF0 автоматизируемого процесса.

На данный момент процесс учета ремонта и обслуживания оборудования и ПО ИТ-инфраструктуры происходит с помощью приложения Telegram и Excel. Сотрудники составляют заявки о проблеме с оборудованием и с указанием номера оборудования присылают заявку в чат Telegram. Обработка с последующим выполнением и учета заявок производится в Excel.

Для автоматизации этого процесса будет разработано приложение, которое объединит в себе функциональность двух используемых приложений, а также даст возможность анализировать выходные данные о заявках, используя машинное обучение технологии ML.NET (о ней будет сказано позднее), и позволит использовать его с любой платформы. Приложение будет содержать в себе:

1. Формирование заявки (номер оборудования, описание проблемы, дата подачи заявки);
2. Просмотр списка заявок, выбор заявок и просмотр содержимого;
3. Обработка заявки (возможность менять статусы заявки на: принято, отклонено, выполнена; указывать дату для осмотра проблемного оборудования; сотрудника, который будет заниматься решением проблемы и комментарий);

4. Формирование отчетности о заявках;
5. Просмотр и редакция планов обслуживания;
6. Анализ отчетов заявок с помощью технологии ML.NET.

Диаграмма вариантов использования представлена на рисунке 3.

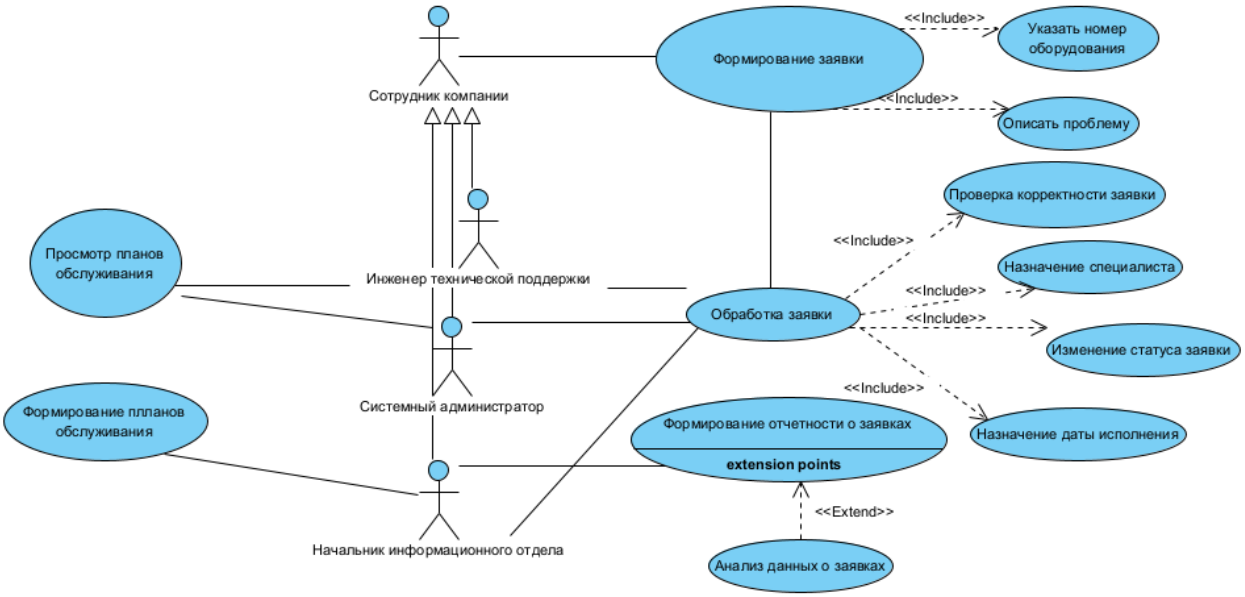

Рисунок 3 – Диаграмма вариантов использования приложения

Для разрабатываемого прототипа приложения разработана база данных на SQL Server. ER-диаграмма базы данных представлена на рисунке 4.

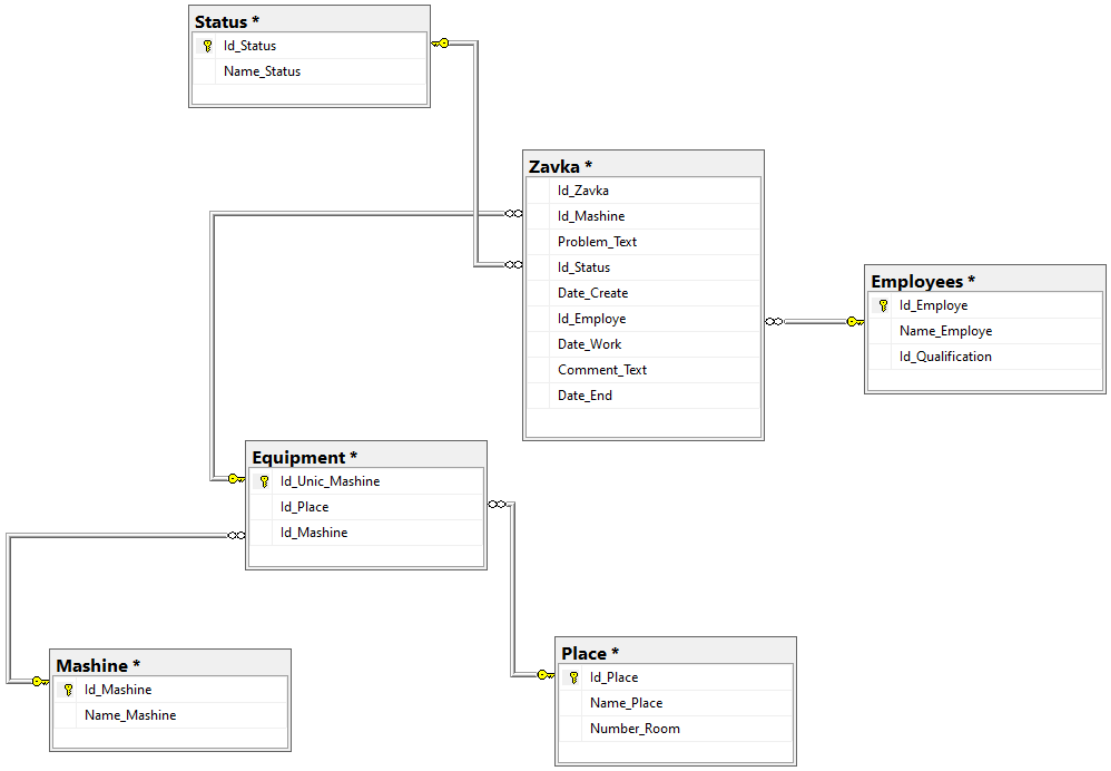

Рисунок 4 – ER-диаграмма базы данных прототипа приложения.

Клиентская часть приложения будет написана на интегрированной среде разработки Visual Studio 2022 с использованием технологии MAUI на .NET. Для демонстрации технологии будет реализован сценарий формирования заявки, который представлен на рисунке 5.

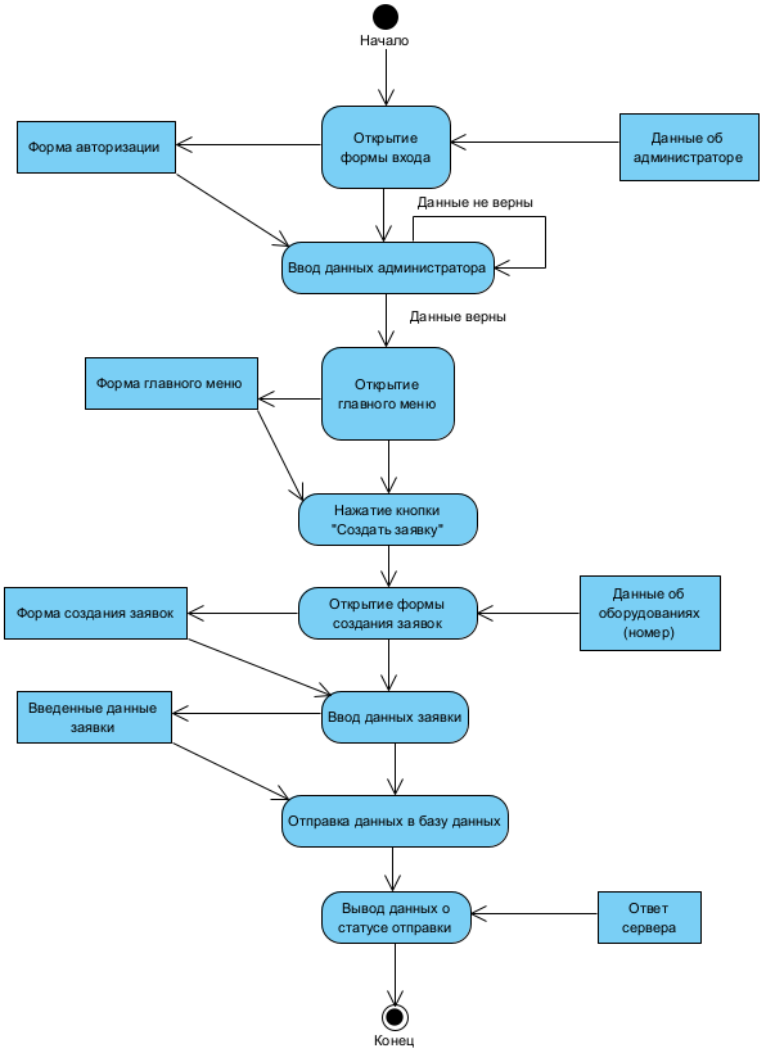

Рисунок 5 – Диаграмма деятельности демонстрационного приложения

Первым делом были созданы 3 страницы – авторизация, главное меню и формирование заявок и организован переход между ними в соответсвии со сценарием на рисунке 5. Запущенное приложение и работа страниц на windows и android представлены на рисунках 6 и 7 соответственно.

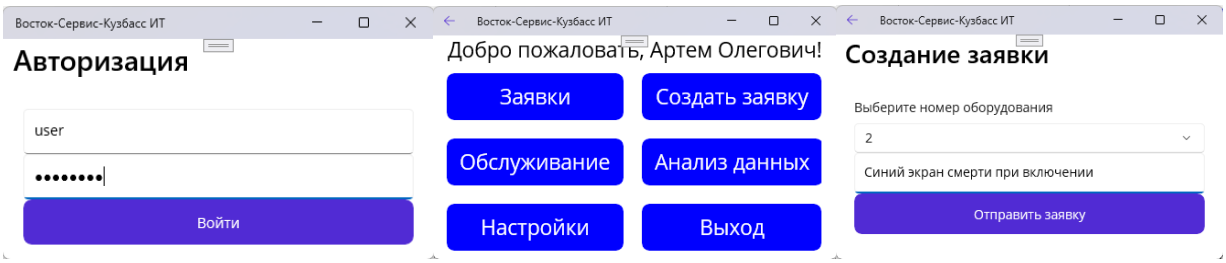

Рисунок 6 – Приложение, запущенное на Windows

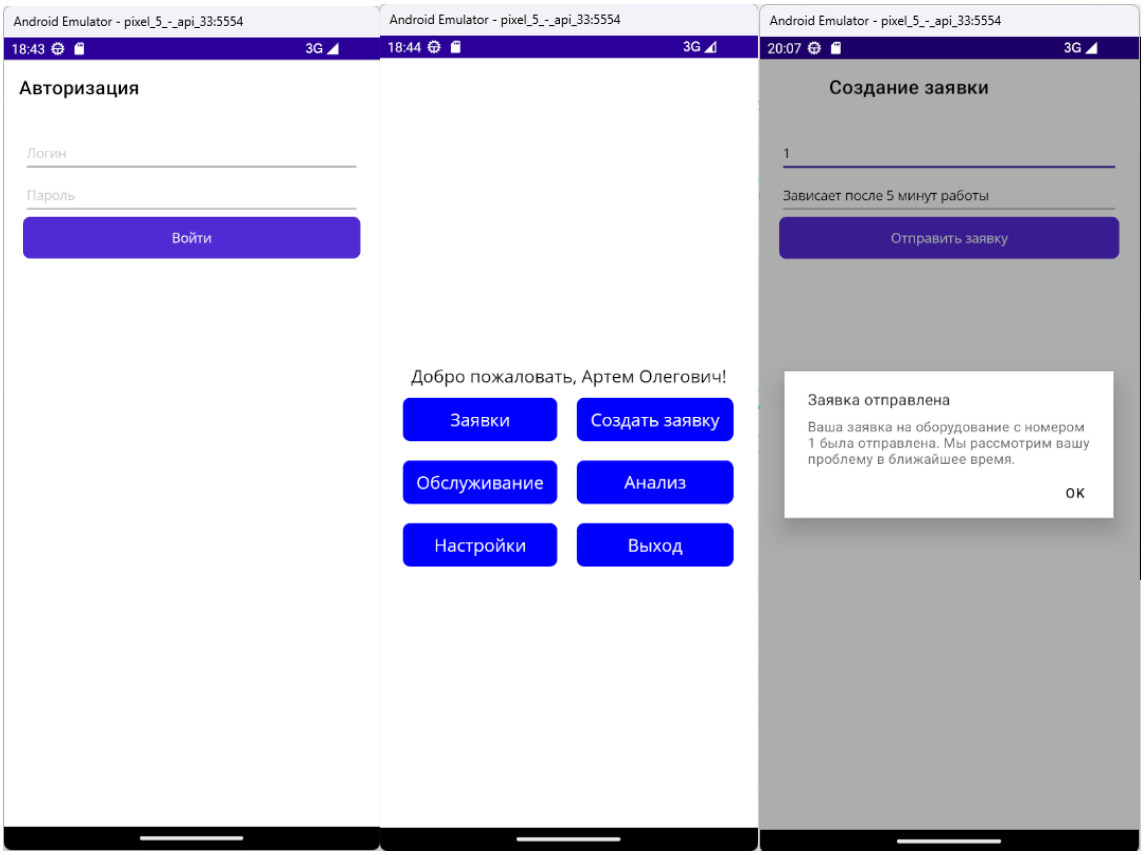

Рисунок 7 – Приложение, запущенное на Android

Для наглядности работы с технологией .NET MAUI код оформления страницы формирования заявки XAML представлен на рисунке 8.

```
<!--Создание самой страницы-->
<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="MauiApp2.CreateRequestPage"
             Title="Создание заявки">
    <!--упорядочивает дочерние представления по горизонтали или по вертикали-->
    <!--Свойство Padding устанавливает отступ-->
    <StackLayout Padding="20">
        <!--Создание выпадающего списка выбора-->
        <Picker x:Name="myPicker" Title="Выберите номер оборудования"/>
        <!--Создание текстового поля ввода-->
        <Entry x:Name="IssueEntry" Placeholder="Описание проблемы" />
        <!--Создание кнопки для вызова обработчика-->
        <Button Text="Отправить заявку" Clicked="SubmitButton_Clicked" />
    </StackLayout>
</ContentPage>
```

Рисунок 8 – Код оформления страницы формирования заявки

Также написан обработчик этой страницы на языке C#, который представлен на рисунке 9. Все действия закомментированы для комфортного чтения кода читателю.

```
public partial class CreateRequestPage : ContentPage
{
    SqlConnection sqlConnection;

    Ссылок: 1
    public CreateRequestPage()
    {
        InitializeComponent(); //инициализация компонент
        //создание строки подключения к базе данных
        string sqlconstr = "Data Source=ARTEMPC;Initial Catalog=IT_Vostok_Service-Kuzbass;Integrated Security=True;TrustServerCertificate=True";
        sqlConnection = new SqlConnection(sqlconstr); //создание подключения к базе данных
        sqlConnection.Open();//открытие соединения с базой данных
        string sqlExpression = "SelectMash"; // создание строки с названием процедуры
        SqlCommand command = new SqlCommand(sqlExpression, sqlConnection); //создание команды для вызова процедуры в выбранном подключении
        command.CommandType = CommandType.StoredProcedure;// указание типа команды, в данный момент это хранимая процедура
        var dataId = command.ExecuteReader(); //выполнение хранимой процедуры с выборкой данных
        ObservableCollection<int> Items = new ObservableCollection<int>(); //создание динамической коллекции данных
        while (dataId.Read())//чтение выборки и добавление ее в коллекцию
        {
            Items.Add((int)dataId["Id_Unic_Mashine"]);
        }
        myPicker.ItemsSource = Items; //привязка коллекции к элементу для выбора номера оборудования
    }
    Ссылок: 0
    private void SubmitButton_Clicked(object sender, EventArgs e) //обработчик кнопки "отправить заявку"
    {
        int equipmentNumber = (int)myPicker.SelectedItem; //получение номера выбранного оборудования
        string issueDescription = IssueEntry.Text; //получение описания проблемы оборудования
        string sqlExpression = "InsertZavka"; // создание строки с названием процедуры
        SqlCommand command = new SqlCommand(sqlExpression, sqlConnection); //создание команды для вызова процедуры в выбранном подключении
        command.CommandType = CommandType.StoredProcedure; // указание типа команды, в данный момент это хранимая процедура
        SqlParameter IdParam = new SqlParameter //создание входного параметра для процедуры
        {
            ParameterName = "@IdUnicMash",
            Value = equipmentNumber
        };
        command.Parameters.Add(IdParam);//добавление параметра для процедуры
        SqlParameter ProblemParam = new SqlParameter//создание входного параметра для процедуры
        {
            ParameterName = "@ProblemText",
            Value = issueDescription
        };
        command.Parameters.Add(ProblemParam);//добавление параметра для процедуры
        command.ExecuteNonQuery();//выполнение процедуры
        //уведомление об отправлении заявки
        DisplayAlert("Заявка отправлена", "Ваша заявка на оборудование с номером " + equipmentNumber +
            " была отправлена. Мы рассмотрим вашу проблему в ближайшее время.", "ОК");
    }
}
```

Рисунок 9 – код обработчки страницы формирования заявки

Для реализации функции анализа данных отчетов заявок будет использована технология ML.NET. Эта технология основана на моделях машинного обучения и позволяет интегрировать и использовать ее в приложениях, основанных на .NET. Упрощенный принцип работы технологии представлен на рисунке 10.

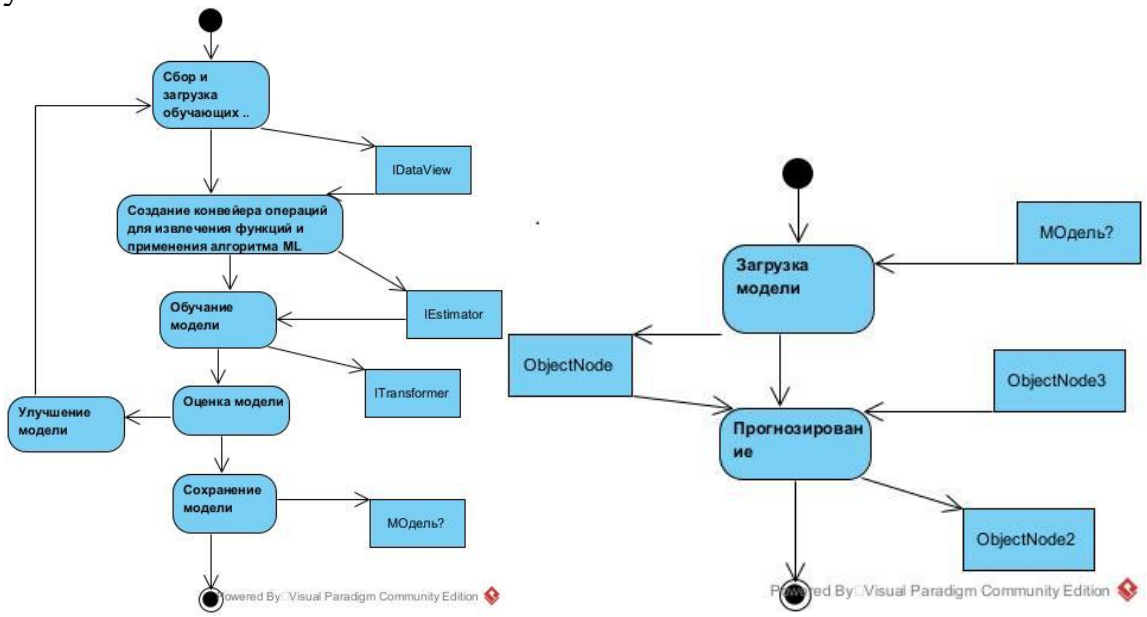

Рисунок 10 – Упрощенный принцип работы технологии ML.NET

Технология ML.NET поддерживает множество задач машинного обучения: двоичную классификацию, классификацию по нескольким классам, регрессию, кластеризацию, обнаружение аномалий, рейтинг (ранжирование), рекомендацию, прогнозирование, классификацию изображений и обнаружение объектов. Для функции исходного процесса выбран анализ на основе модели регрессии дерева принятия решений, которая позволит предсказать примерное время на выполнение заявок на основе уже известных данных.

Набор данных будет содержать следующие столбцы:

- Mashin_id – идентификатор оборудования (признак)
- Problem_id – идентификатор проблемы (признак)
- Specialist_id – идентификатор специалиста (признак)
- Time_create – время создания заявки (признак)
- Time_work – время, назначенное на выполнение заявки (признак)
- Time_remont – время, затраченное на выполнение заявки (метка – прогнозируемое значение. Разница времени, назначенного на ремонт и выполнения заявки)

Для обучения модели и прогнозирования нужно:

1. Подключить библиотеку Microsoft.ML;
2. Подготовить и проанализировать данные (сделано выше);
3. Создать классы для входных данных и для результата прогнозирования.
4. Определить пути к данным для обучения и оценки, и путь к модели для сохранения;
5. Инициализировать переменную ML контекста и метод обучения Train;
6. Загрузка и преобразование данных (ML.NET использует DataView, который аналогичен DataFrame в библиотеке Pandas для Python);
7. Выбрать алгоритм для обучения (для данного случая будет выбран FastTreeRegressionTrainer – регрессия дерева принятия решений);
8. Обучить модель с помощью метода Fit, где входными данными будут преобразованные данные из п. 6;
9. Оценить модель с помощью тестовых данных, определенных в п. 4;
10. Использовать модель для прогнозирования (создать новый класс с входными данными и передать в функцию Predict()).

На данный момент эта функция не реализована, т.к. еще не были получены необходимые данные.

Список литературы:

1. Моделирование на языке UML в среде Visual Paradigm 14. [Электронный ресурс] – URL: http://sp.cs.msu.ru/ooap/exer2017.html#exer34 (дата обращения: 26.03.2024).
2. Документация по ML.NET. [Электронный ресурс] – URL: https://learn.microsoft.com/ru-ru/dotnet/machine-learning/ (дата обращения: 26.03.2024).
3. .NET MAUI. [Электронный ресурс] – URL: https://learn.microsoft.com/ru-ru/dotnet/maui/?view=net-maui-8.0 (дата обращения: 26.03.2024).
4. Понимание XAML. [Электронный ресурс] – URL: https://habr.com/ru/articles/141069/ (дата обращения: 26.03.2024).
5. .NET MAUI и C# создание кроссплатформенных приложений. [Электронный ресурс] – URL: https://metanit.com/sharp/maui/1.1.php (дата обращения: 26.03.2024).