

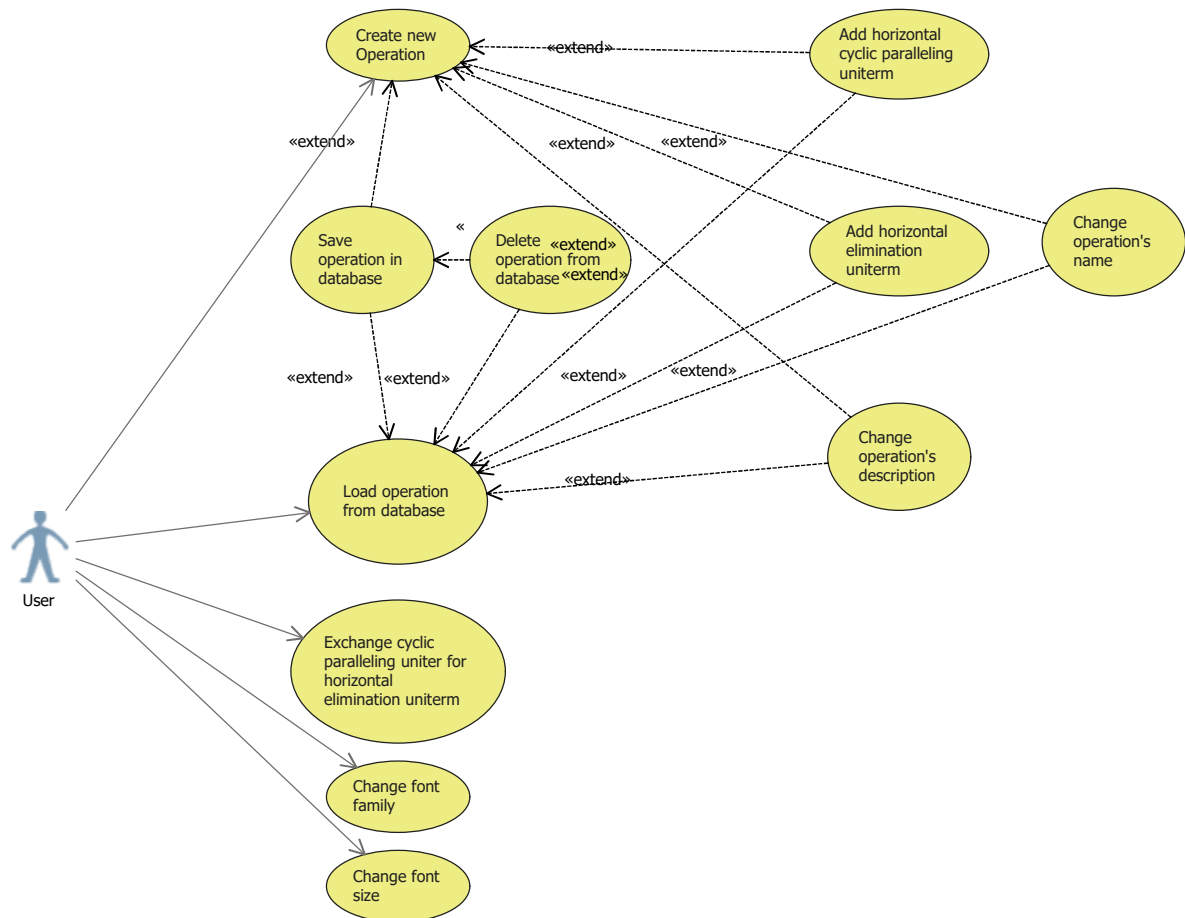
<p align="center">Politechnika Świętokrzyska Wydział Elektrotechniki, Automatyki i Informatyki Zakład Zastosowań Informatyki</p>	
<p align="center">Modelowanie i analiza systemów informatycznych – Projekt</p>	
<p align="center">Temat: Zamiana unitermu poziomej operacji cyklicznego zrównoleglenia unitermów na poziomą operację eliminowania unitermów.</p>	<p align="center">Adrian Jakubczyk 1ID21A</p>

1. Wykorzystane technologie.

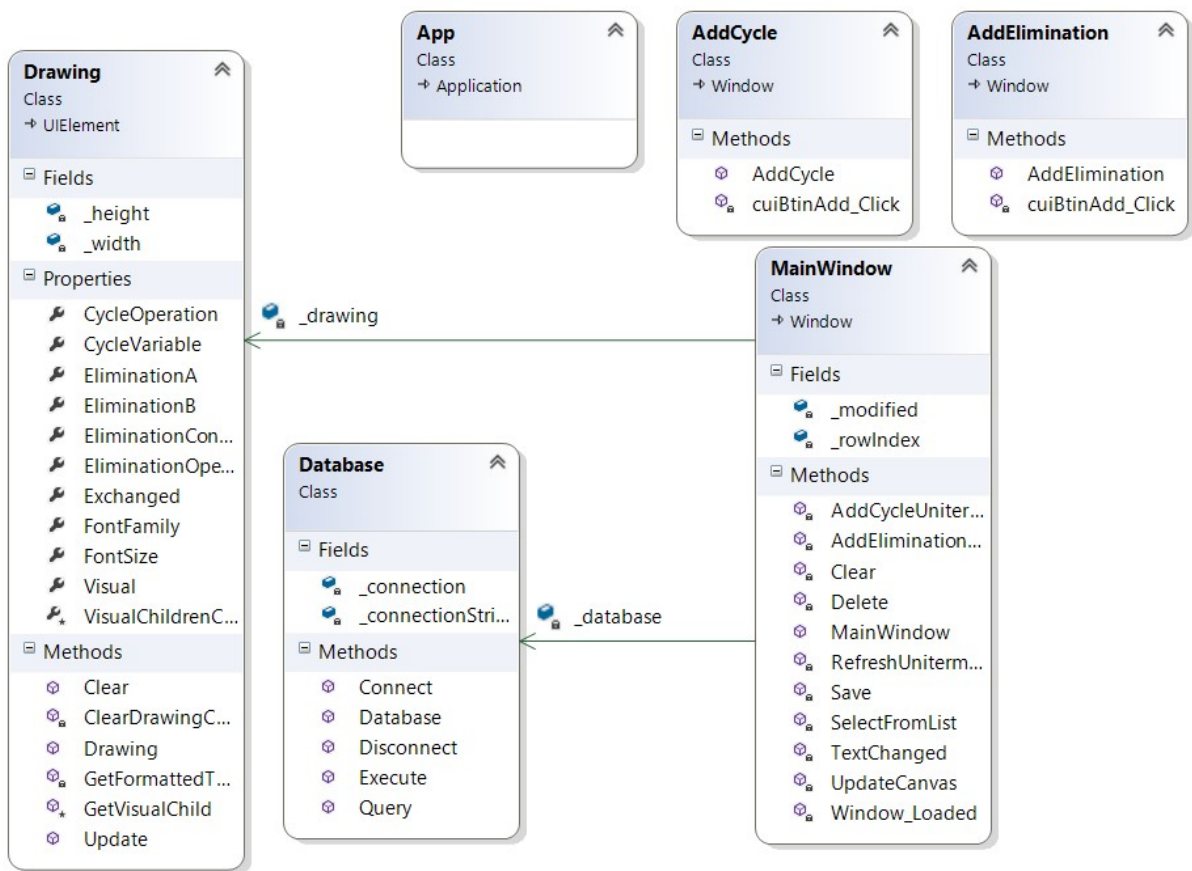
- Visual Studio 2019 Community - zintegrowane środowisko programistyczne firmy Microsoft. Jest używane do tworzenia oprogramowania konsolowego oraz z graficznym interfejsem użytkownika, w tym aplikacji Windows Forms, WPF, Web Sites, Web Applications i inne. Aplikacje mogą być pisane na platformy: Microsoft Windows, Windows Phone, Windows CE, .NET Framework, Microsoft Silverlight, Linux, MacOS oraz konsole XBOX.
- .NET Framework - platforma programistyczna opracowana przez Microsoft, obejmująca środowisko uruchomieniowe (Common Language Runtime – CLR) oraz biblioteki klas dostarczające standardowej funkcjonalności dla aplikacji. Technologia ta nie jest związana z żadnym konkretnym językiem programowania, a programy mogą być pisane w jednym z wielu języków – na przykład C++/CLI, C#, F#, J#, Delphi 8 dla .NET, Visual Basic .NET. Zadaniem platformy .NET Framework jest zarządzanie różnymi elementami systemu: kodem aplikacji, pamięcią i zabezpieczeniami.
- WPF - Windows Presentation Foundation, nazwa silnika graficznego i API bazującego na .NET 3, wchodzącego w skład WinFX. WPF integruje interfejs użytkownika, grafikę 2D i 3D, multimedia, oraz dokumenty. API w WPF opiera się na języku XML, dokładniej na jego implementacji o nazwie XAML.
- MSSQL - Microsoft SQL Server, system zarządzania bazą danych, wspierany i rozpowszechniany przez korporację Microsoft. Jest to główny produkt bazodanowy tej firmy, który charakteryzuje się tym, iż jako język zapytań używany jest przede wszystkim Transact-SQL, który stanowi rozwinięcie standardu ANSI/ISO.

2. Modelowanie i diagramy UML

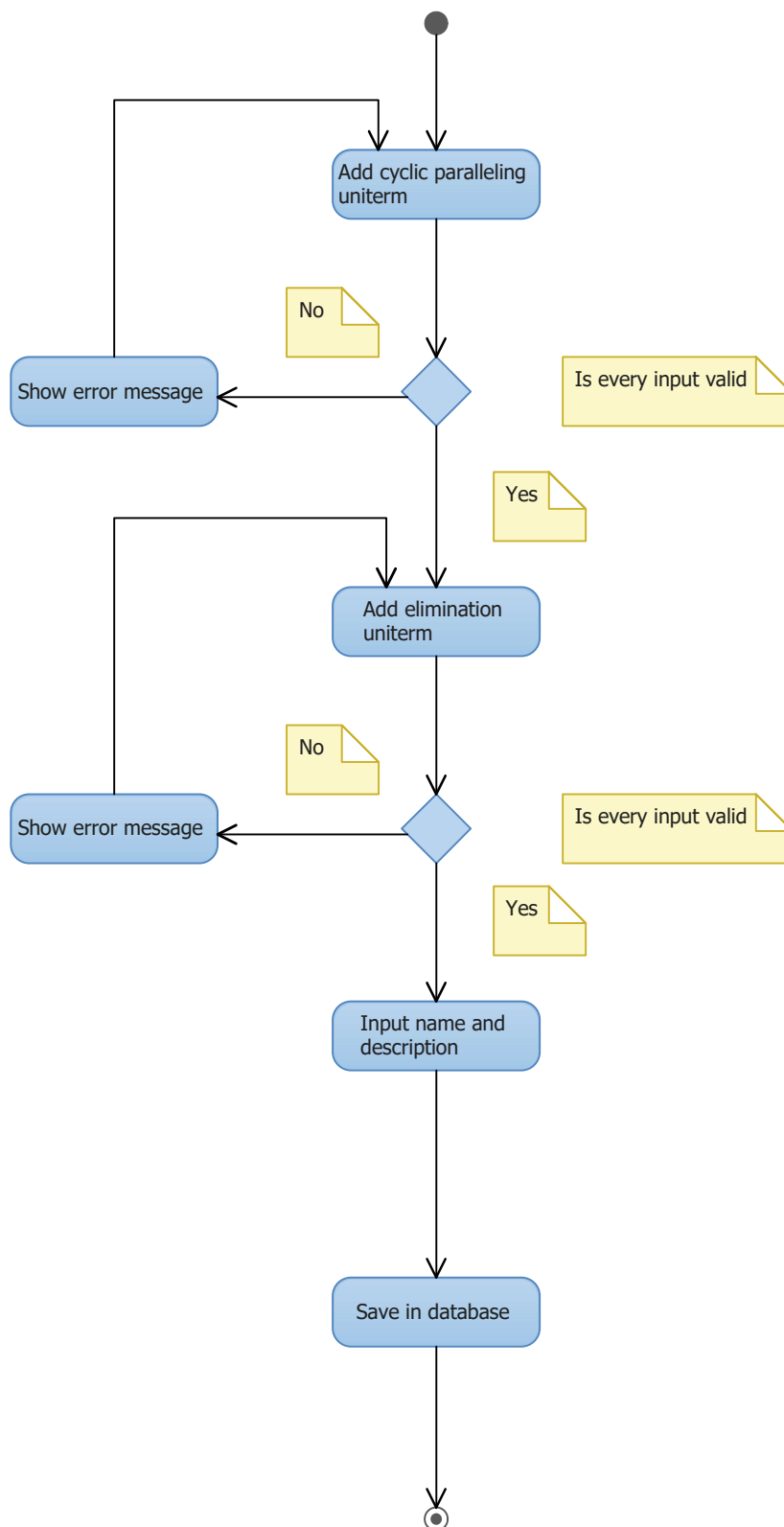
2.1. Diagram przypadków użycia.



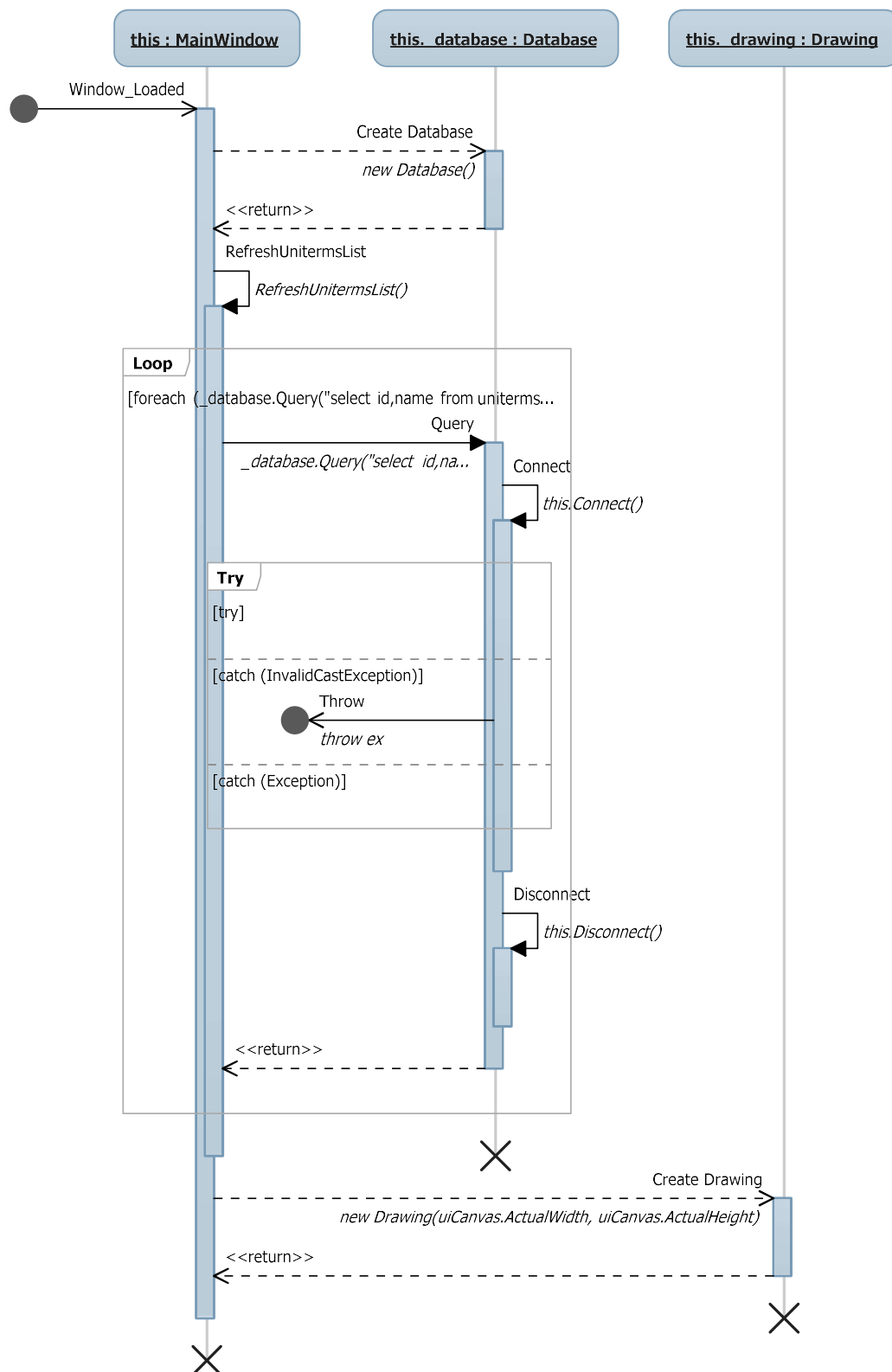
2.2. Diagram klas.

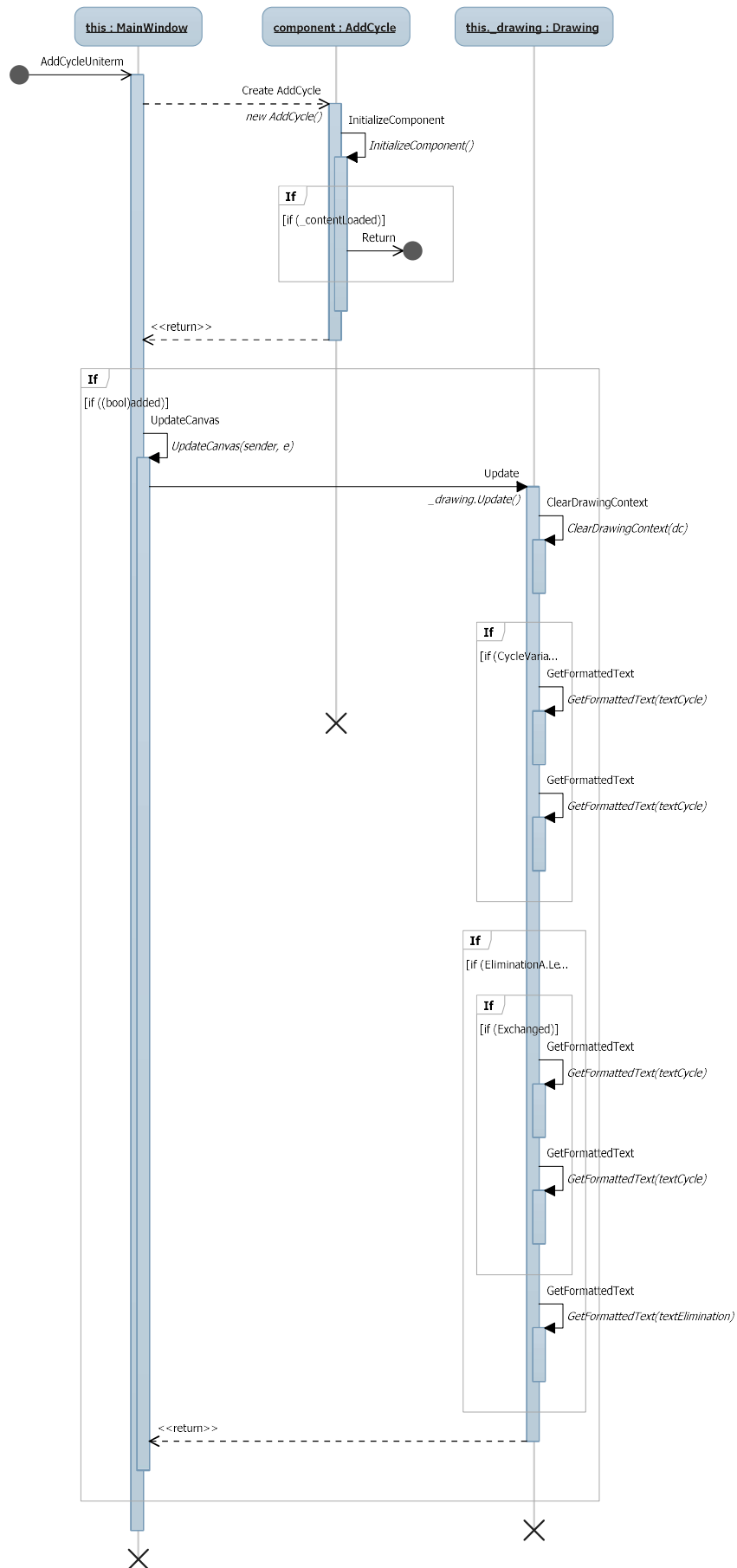


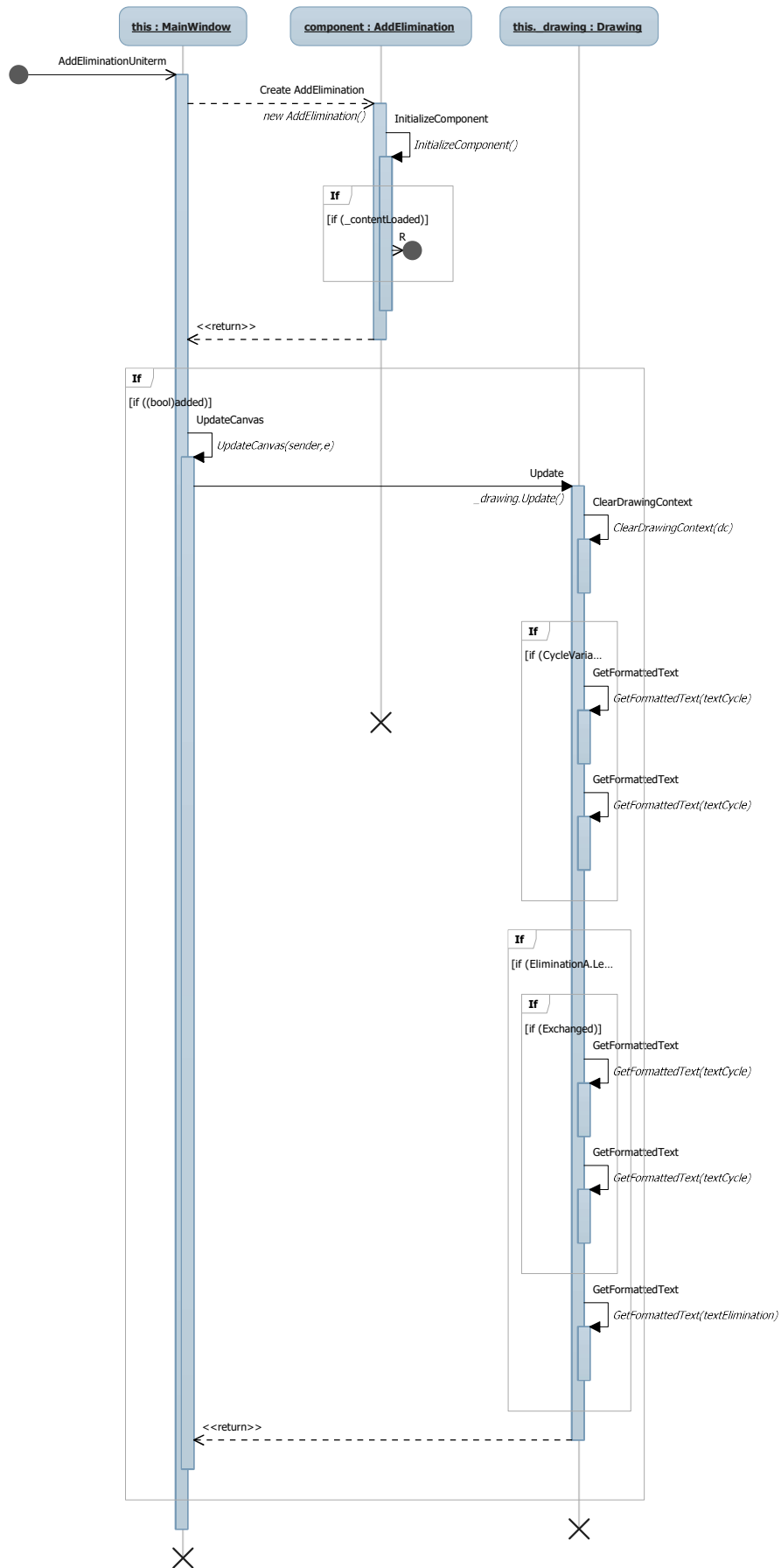
2.3. Diagram aktywności.

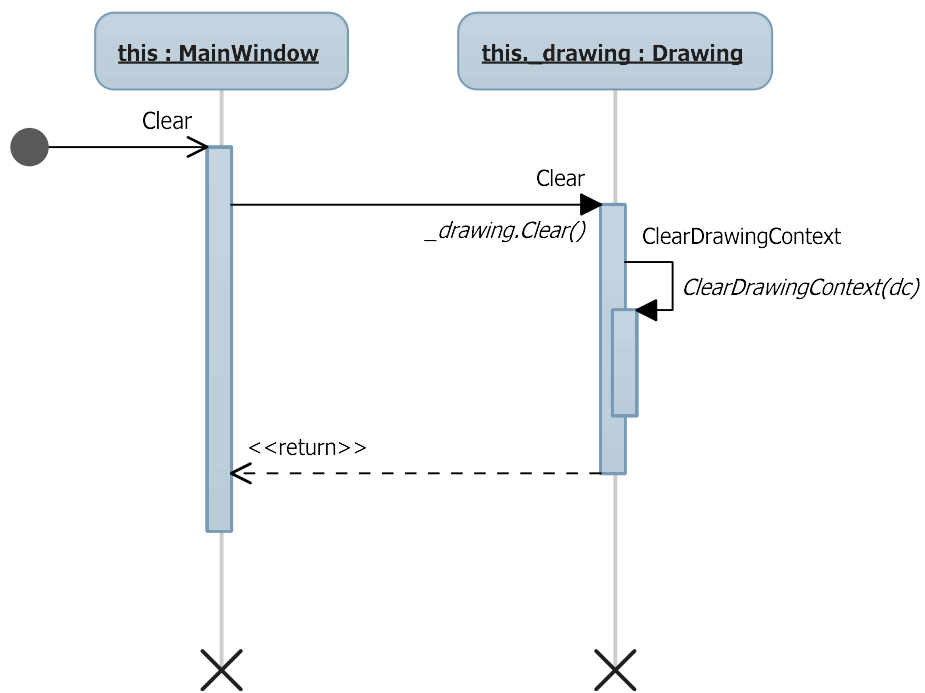


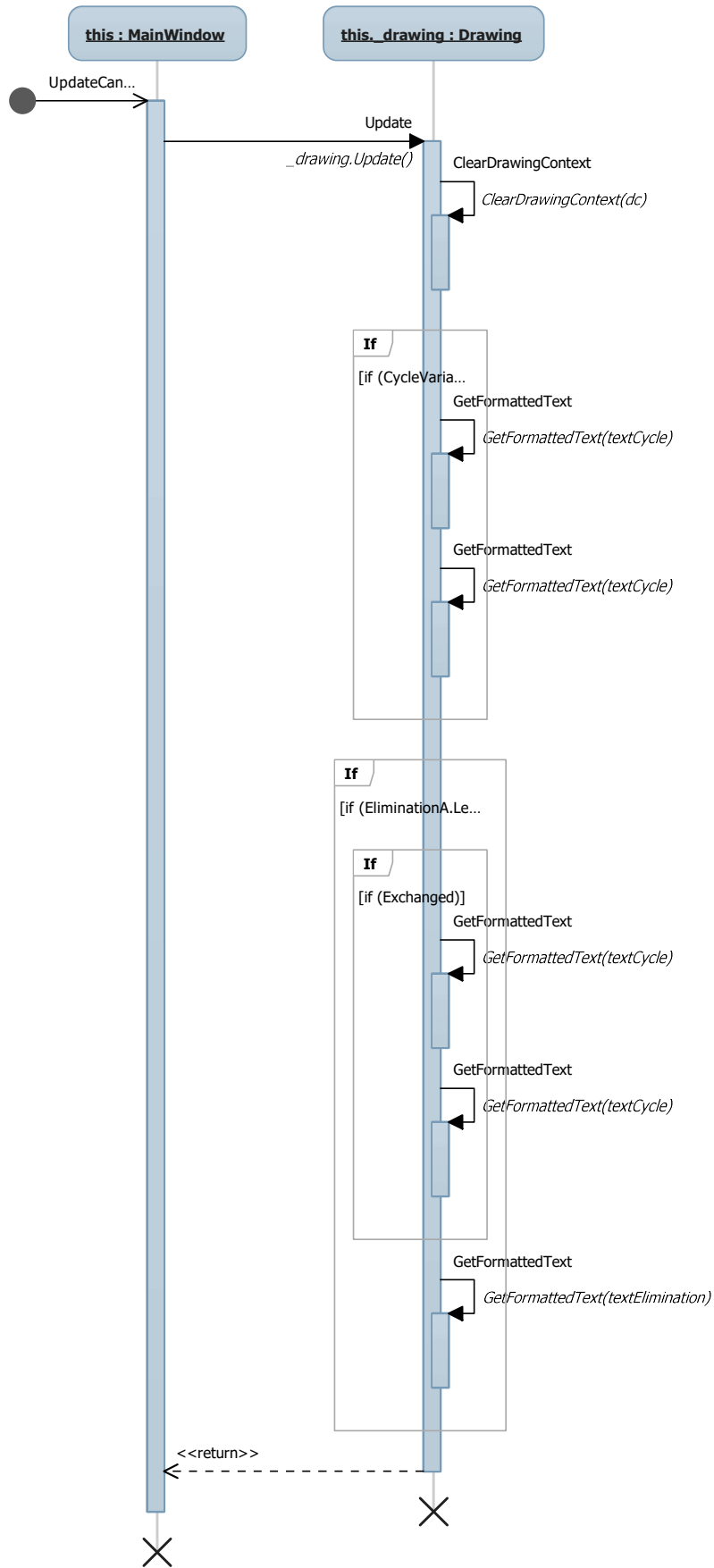
2.4. Diagramy sekwencji.

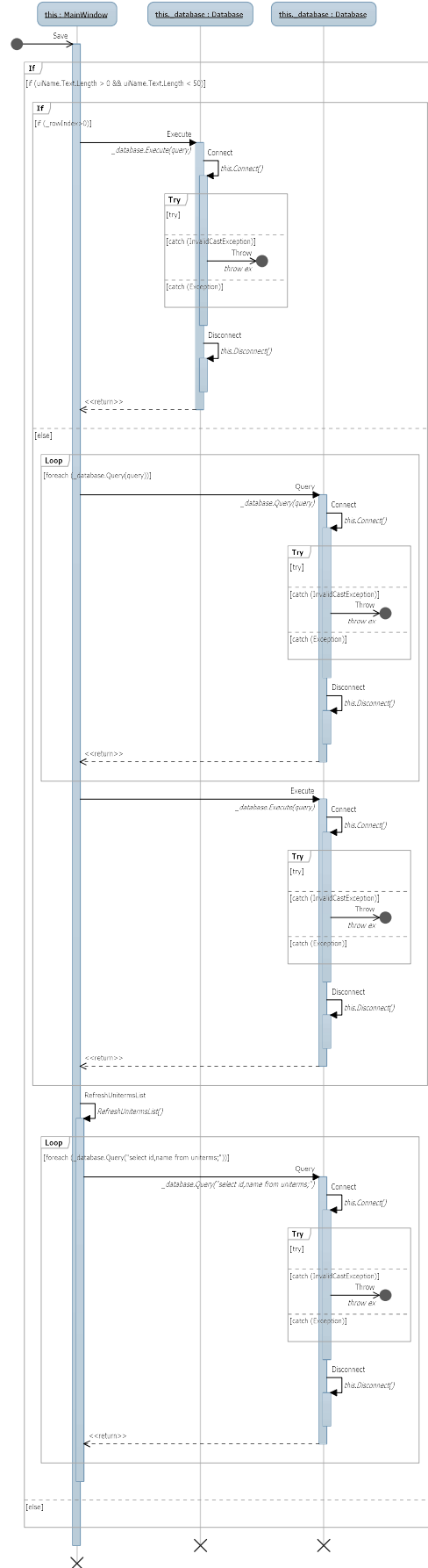


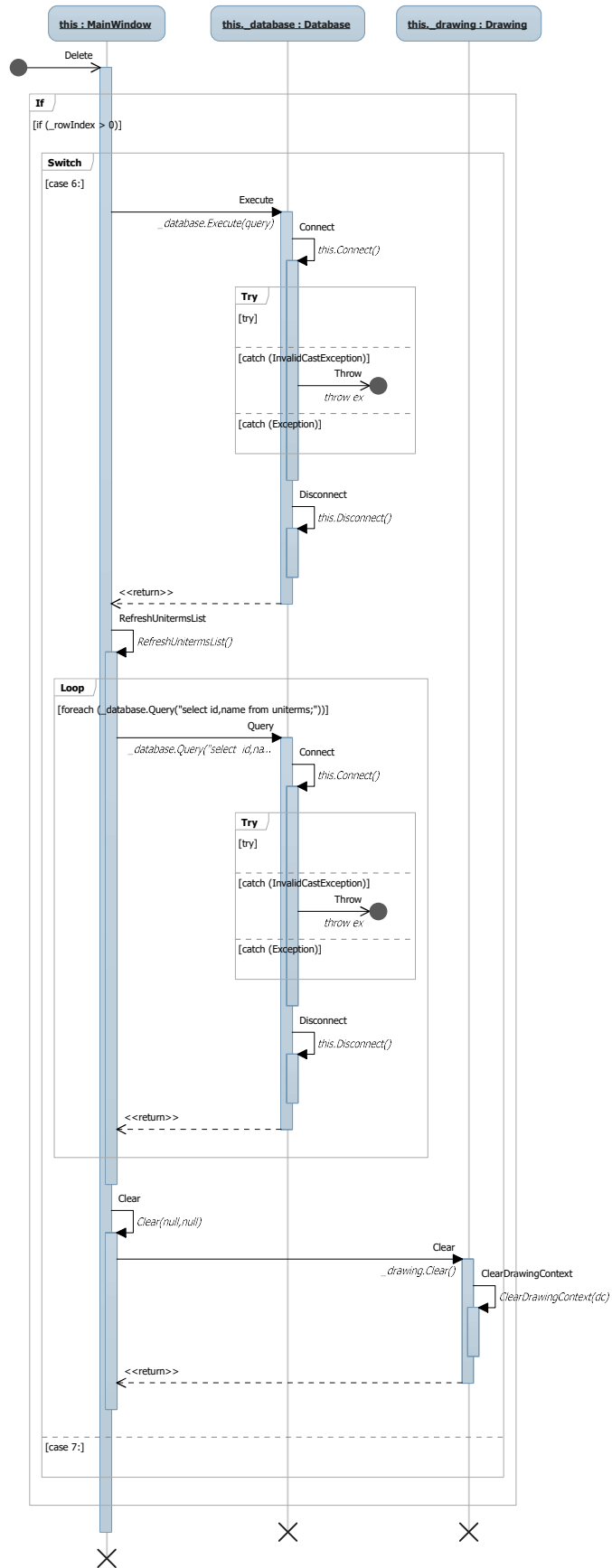


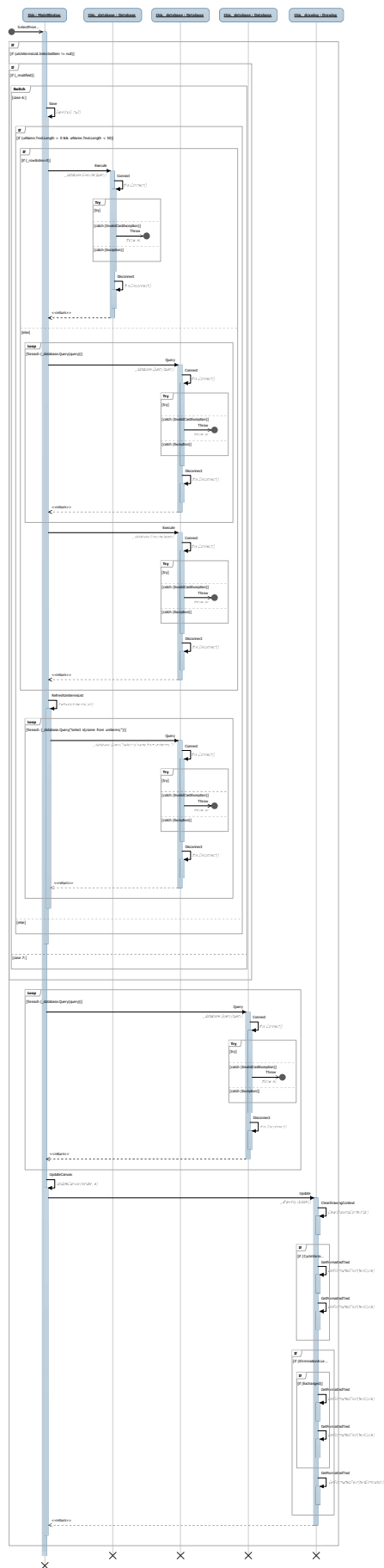


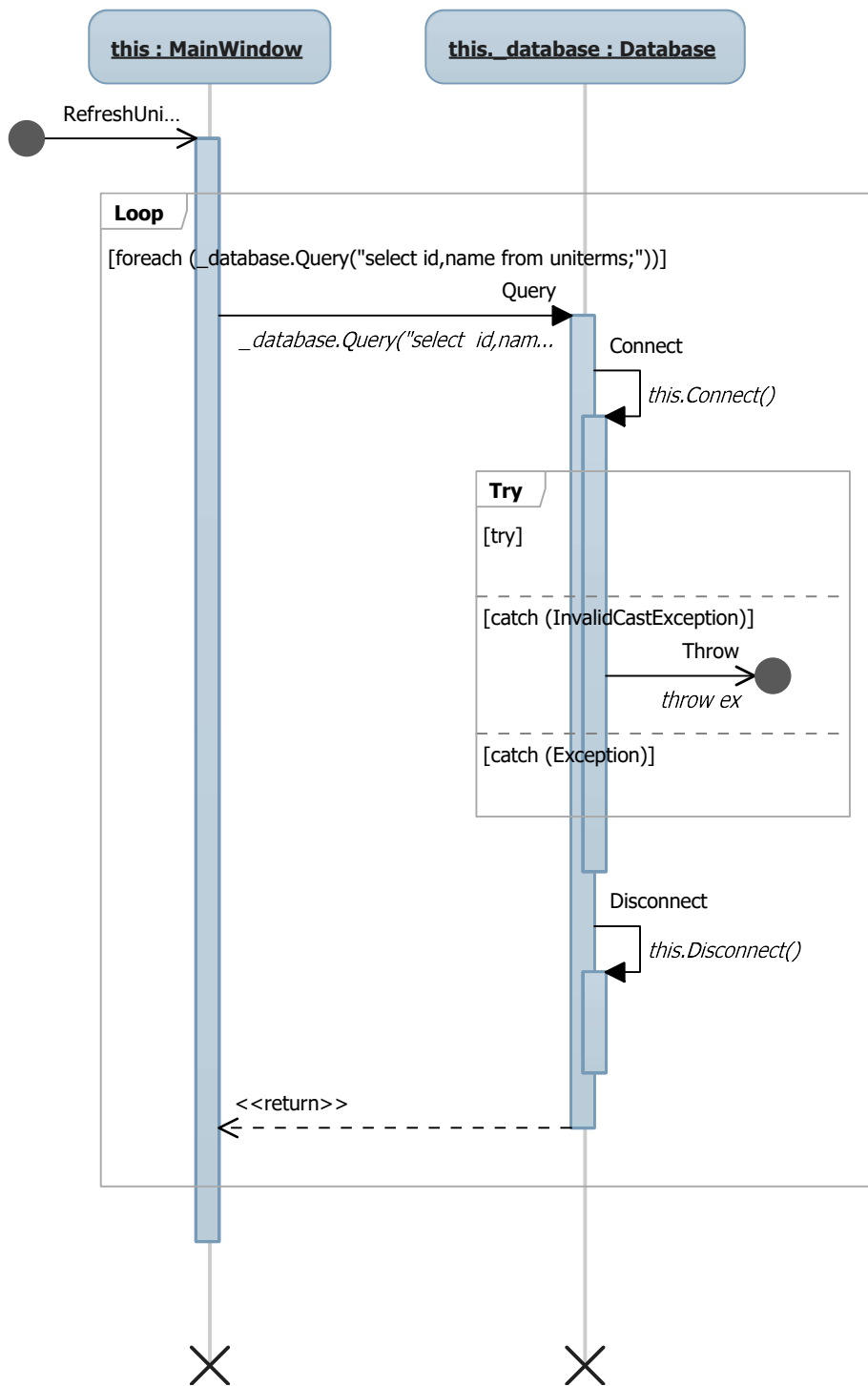




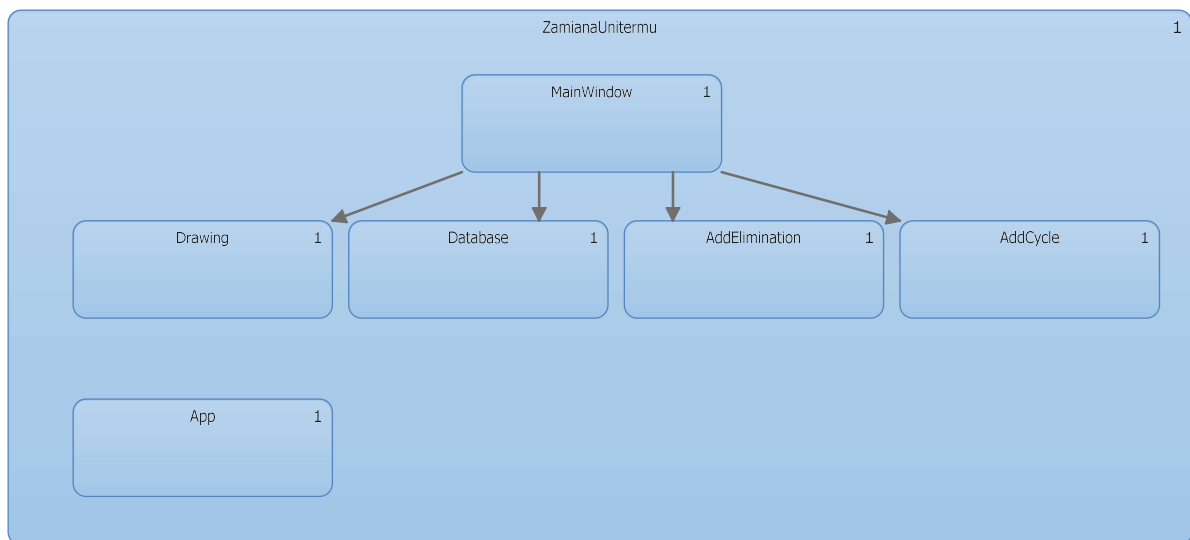




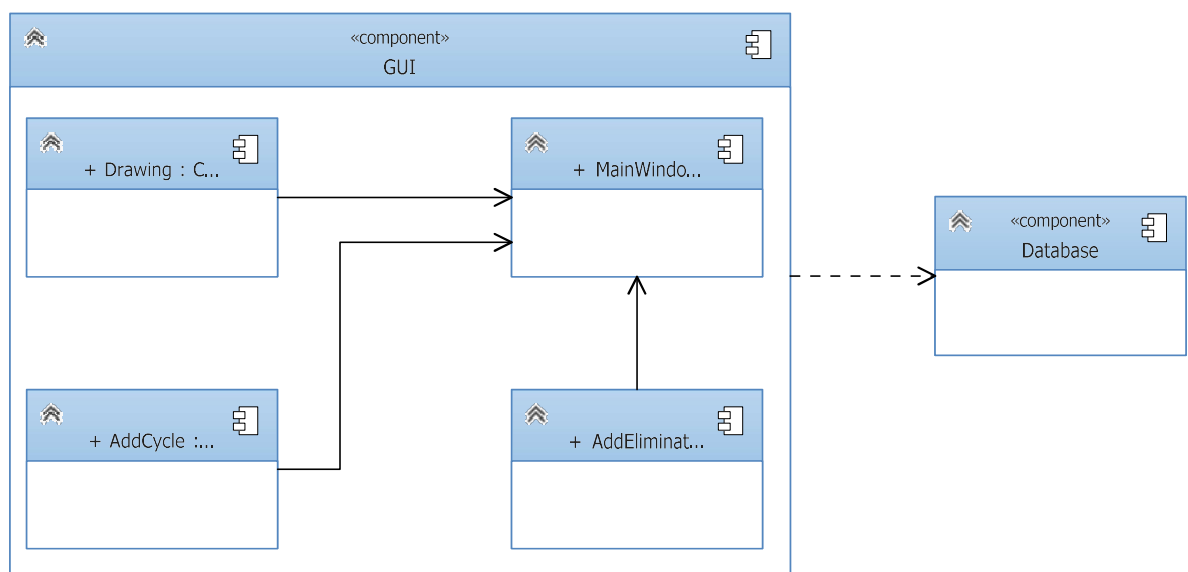




2.5. Diagram warstw.

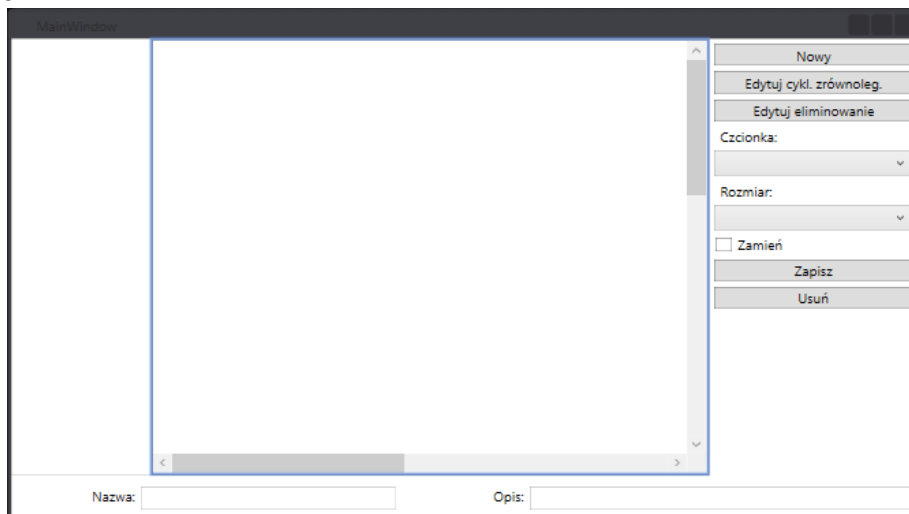


2.6. Diagram komponentów.



3. Analiza kodu źródłowego.

Główne okno:



```
<Window x:Class="ZamianaUnitermu.MainWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:ZamianaUnitermu"
    mc:Ignorable="d"
    Title="MainWindow" Height="450" Width="800"
    ResizeMode="CanMinimize"
    WindowStartupLocation="CenterScreen"
    Loaded="Window_Loaded">

    <Grid ClipToBounds="True">
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="10*" />
            <ColumnDefinition Width="40*" />
            <ColumnDefinition Width="15*" />
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition Height="150*" />
            <RowDefinition Height="15*" />
        </Grid.RowDefinitions>

        <ScrollViewer Grid.Row="0" Grid.Column="0" BorderThickness="2"
            VerticalScrollBarVisibility="Auto">
            <ListBox x:Name="uiUnitermsList" />
        </ScrollViewer>

        <StackPanel Grid.Row="0" Grid.Column="2" Orientation="Vertical" Margin="5,0,5,0">
            <Button x:Name="uiBtnNew" Content="Nowy" Margin="0,5,0,0"/>
            <Button x:Name="uiBtnCycle" Content="Edytuj cykl. równoleg." Margin="0,5,0,0"/>
            <Button x:Name="uiBtnElimination" Content="Edytuj eliminowanie" Margin="0,5,0,0"/>
            <Label Content="Czcionka:" />
            <ComboBox x:Name="uiFontFamily" />
            <Label Content="Rozmiar:" />
            <ComboBox x:Name="uiFontSize" />
            <CheckBox x:Name="uiExchange" Content="Zamień" IsChecked="False" Margin="0,5,0,0" />
            <Button x:Name="uiBtnSave" Content="Zapisz" Margin="0,5,0,0"/>
            <Button x:Name="uiBtnDelete" Content="Usuń" Margin="0,5,0,0"/>
        </StackPanel>

        <Border BorderThickness="2" BorderBrush="Gray" Grid.Column="1" Grid.Row="0">
            <ScrollViewer HorizontalScrollBarVisibility="Auto" ClipToBounds="True">
                <Canvas x:Name="uiCanvas" Width="1000" Height="1000"/>
            </ScrollViewer>
        </Border>
    </Grid>
```

```

<Grid Grid.Row="1" Grid.ColumnSpan="3" Margin="0,5,0,5">
    <Grid.ColumnDefinitions>
        <ColumnDefinition Width="1*"/>
        <ColumnDefinition Width="2*"/>
        <ColumnDefinition Width="1*"/>
        <ColumnDefinition Width="3*"/>
    </Grid.ColumnDefinitions>
    <Label Grid.Column="0" Content="Nazwa:" HorizontalAlignment="Right"/>
    <TextBox x:Name="uiName" Grid.Column="1" Height="20" Margin="0,0,5,0"/>
    <Label Grid.Column="2" Content="Opis:" HorizontalAlignment="Right"/>
    <TextBox x:Name="uiDescription" Grid.Column="3" Height="20" Margin="0,0,5,0"/>
</Grid>

</Grid>

</Window>

```

Metoda Window_Loaded:

```

private void Window_Loaded(object sender, RoutedEventArgs e)
{
    foreach (FontFamily f in Fonts.SystemFontFamilies)
    {
        uiFontFamily.Items.Add(f);
    }
    if (uiFontFamily.Items.Count > 0)
        uiFontFamily.SelectedIndex = uiFontFamily.Items.IndexOf(new FontFamily("Arial"));

    for (int i = 8; i <= 40; i++)
    {
        uiFontSize.Items.Add(i);
    }
    uiFontSize.SelectedIndex = 4;

    _database = new Database();

    RefreshUnitermsList();

    _drawing = new Drawing(uiCanvas.ActualWidth, uiCanvas.ActualHeight);
    _drawing.FontFamily = new FontFamily(uiFontFamily.SelectedItem.ToString());
    _drawing.FontSize = Int32.Parse(uiFontSize.SelectedItem.ToString());
    uiCanvas.Children.Add(_drawing);

    uiBtnNew.Click += Clear;
    uiBtnCycle.Click += AddCycleUniterm;
    uiBtnElimination.Click += AddEliminationUniterm;
    uiBtnSave.Click += Save;
    uiBtnDelete.Click += Delete;

    uiExchange.Click += UpdateCanvas;
    uiFontFamily.SelectionChanged += UpdateCanvas;
    uiFontSize.SelectionChanged += UpdateCanvas;
    uiUnitermsList.SelectionChanged += SelectFromList;

    uiName.TextChanged += TextChanged;
    uiDescription.TextChanged += TextChanged;

    _rowIndex = -1;
    _modified = false;
}

```

Po załadowaniu okna ładowane są czcionki dostępne w systemie i wstawiane w ComboBox. Następnie Ustawiana jest domyślna czcionka Arial. Dodawane są rozmiary czcionki, tworzony jest obiekt bazy danych i ładowane są dostępne unitermy. Później tworzona jest powierzchnia do rysowania, a następnie do przycisków i kontrolki przypisywane są odpowiednie funkcje.

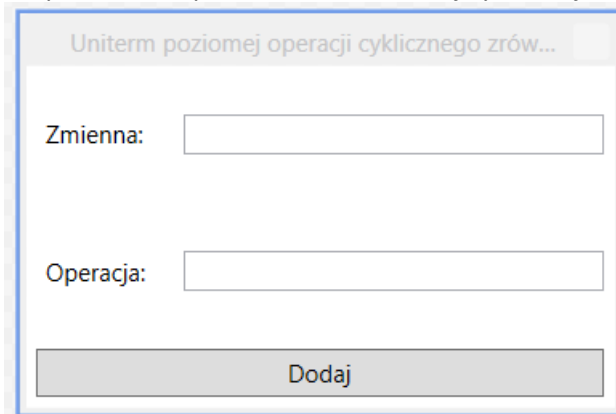
Metoda AddCycleUniterm:

```
private void AddCycleUniterm(object sender, RoutedEventArgs e)
{
    AddCycle addCycleWindow = new AddCycle();

    bool? added = addCycleWindow.ShowDialog();
    if ((bool)added)
    {
        string variable = addCycleWindow.cuiVariable.Text;
        if (variable.Length < 50 && variable.Length > 0)
        {
            _drawing.CycleVariable = variable;
        }
        else
        {
            MessageBox.Show("Zmienna: nie podano lub jest za długa");
        }

        string operation = addCycleWindow.cuiOperation.Text;
        if (operation.Length < 50 && operation.Length > 0)
        {
            _drawing.CycleOperation = operation;
        }
        else
        {
            MessageBox.Show("Operacja: nie podano lub jest za długa");
        }
        UpdateCanvas(sender, e);
        _modified = true;
    }
}
```

Tworzone jest nowe okno z polami do wprowadzania zmiennej cyklicznej i operacji:



Uniterm poziomej operacji cyklicznego zrów...

Zmienna:

Operacja:

Metoda AddEliminationUniterm:

```
private void AddEliminationUniterm(object sender, RoutedEventArgs e)
{
    AddElimination addEliminationWindow = new AddElimination();

    bool? added = addEliminationWindow.ShowDialog();
    if ((bool)added)
    {
        string eliminationA = addEliminationWindow.euiA.Text;
        if (eliminationA.Length < 50 && eliminationA.Length > 0)
        {
            _drawing.EliminationA = eliminationA;
        }
        else
        {
            MessageBox.Show("A: nie podano lub jest za długa");
        }

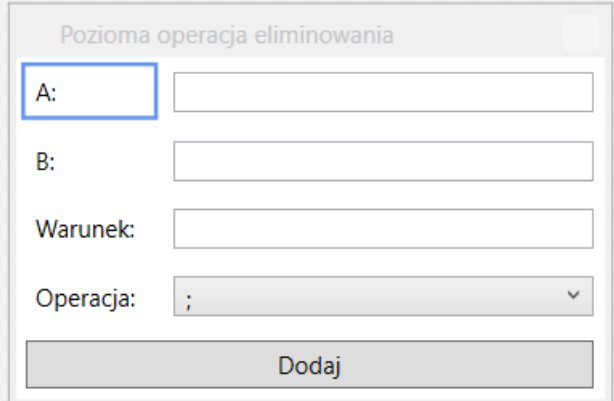
        string eliminationB = addEliminationWindow.euiB.Text;
        if (eliminationB.Length < 50 && eliminationB.Length > 0)
        {
            _drawing.EliminationB = eliminationB;
        }
        else
        {
            MessageBox.Show("B: nie podano lub jest za długa");
        }

        string eliminationCondition = addEliminationWindow.euiCondition.Text;
        if (eliminationCondition.Length < 50 && eliminationCondition.Length > 0)
        {
            _drawing.EliminationCondition = eliminationCondition;
        }
        else
        {
            MessageBox.Show("Warunek: nie podano lub jest za długi");
        }

        _drawing.EliminationOperation = addEliminationWindow.euiOperation.Text;

        UpdateCanvas(sender,e);
        _modified = true;
    }
}
```

Tworzone jest nowe okno do wprowadzenia unitermu eliminacji:



Pozioma operacja eliminowania

A:

B:

Warunek:

Operacja:

Metoda Clear:

```
private void Clear(object sender, RoutedEventArgs e)
{
    uiCanvas.Children.Remove(_drawing);
    _drawing.CycleVariable = "";
    _drawing.CycleOperation = "";
    _drawing.EliminationA = "";
    _drawing.EliminationB = "";
    _drawing.EliminationCondition = "";
    _drawing.EliminationOperation = "";
    _drawing.Clear();
    _rowIndex = -1;
    _modified = false;
    uiName.Text = "";
    uiDescription.Text = "";
    uiCanvas.Children.Add(_drawing);
}
```

Czyści przestrzeń rysowniczą i wprowadzone dane unitermów.

Metoda UpdateCanvas:

```
private void UpdateCanvas(object sender, RoutedEventArgs e)
{
    uiCanvas.Children.Remove(_drawing);

    _drawing.FontFamily = new FontFamily(uiFontFamily.SelectedItem.ToString());
    _drawing.FontSize = Int32.Parse(uiFontSize.SelectedItem.ToString());
    _drawing.Exchanged = (bool)uiExchange.IsChecked;

    _drawing.Update();

    uiCanvas.Children.Add(_drawing);
}
```

Służy do przerysowania unitermów po każdej zmianie danych.

Metoda Save:

```
private void Save(object sender, RoutedEventArgs e)
{
    if (uiName.Text.Length > 0 && uiName.Text.Length < 50)
    {
        if (_rowIndex > 0)
        {
            string query = "UPDATE uniterms SET " +
                "variable='" + _drawing.CycleVariable + "', " +
                "operation='" + _drawing.CycleOperation + "', " +
                "eA='" + _drawing.EliminationA + "', " +
                "eB='" + _drawing.EliminationB + "', " +
                "eCondition='" + _drawing.EliminationCondition + "', " +
                "eOperation='" + _drawing.EliminationOperation + "', " +
                "name='" + uiName.Text + "', " +
                "description='" + uiDescription.Text + "' " +
                "WHERE id='" + _rowIndex + "';";

            _database.Execute(query);

        }
        else
        {
            string query = "SELECT IDENT_CURRENT('uniterms')+1 as id;";
            foreach (DataRow dataRow in _database.Query(query))
            {
                _rowIndex = Int32.Parse(dataRow["id"].ToString());
            }

            query = "INSERT INTO uniterms
(variable,operation,eA,eB,eCondition,eOperation,name,description) VALUES(' " +
                _drawing.CycleVariable + "', '" +
                _drawing.CycleOperation + "', '" +
                _drawing.EliminationA + "', '" +
                _drawing.EliminationB + "', '" +
                _drawing.EliminationCondition + "', '" +
                _drawing.EliminationOperation + "', '" +
                uiName.Text + "', '" +
                uiDescription.Text + "')";

            _database.Execute(query);

        }
        _modified = false;
        RefreshUnitermsList();
    }
    else
    {
        MessageBox.Show("Błędna nazwa");
    }
}
```

Służy do zapisania operacji lub zmian

Metoda Delete:

```
private void Delete(object sender, RoutedEventArgs e)
{
    if (_rowIndex > 0)
    {
        switch (MessageBox.Show("Czy na pewno chcesz usunąć?", "Usuwanie",
        MessageBoxButton.YesNoCancel, MessageBoxImage.Question))
        {
            case MessageBoxResult.Yes:
            {
                string query = "DELETE FROM uniterms WHERE id=" + _rowIndex;
                _database.Execute(query);
                RefreshUnitermsList();
                Clear(null, null);
                break;
            }
            case MessageBoxResult.No:
            case MessageBoxResult.Cancel:
            default: break;
        }
    }
}
```

Służy do usunięcia wybranej operacji z bazy danych.

Metoda SelectFromList:

```
private void SelectFromList(object sender, RoutedEventArgs e)
{
    if (uiUnitermsList.SelectedItem != null)
    {
        string query = "SELECT * FROM uniterms WHERE id=" +
        uiUnitermsList.SelectedItem.ToString().Split(' ')[0];

        if (_modified)
        {
            switch (MessageBox.Show("Czy chcesz zapisać zmiany?", "Zapis",
            MessageBoxButton.YesNoCancel, MessageBoxImage.Question))
            {
                case MessageBoxResult.Yes:
                {
                    Save(null, null);
                    break;
                }
                case MessageBoxResult.No:
                case MessageBoxResult.Cancel:
                default: break;
            }
        }

        foreach (DataRow dataRow in _database.Query(query))
        {
            _drawing.CycleVariable = dataRow["variable"].ToString();
            _drawing.CycleOperation = dataRow["operation"].ToString();
            _drawing.EliminationA = dataRow["eA"].ToString();
            _drawing.EliminationB = dataRow["eB"].ToString();
            _drawing.EliminationCondition = dataRow["eCondition"].ToString();
            _drawing.EliminationOperation = dataRow["eOperation"].ToString();
            _rowIndex = Int32.Parse(dataRow["id"].ToString());
            uiName.Text = dataRow["name"].ToString();
            uiDescription.Text = dataRow["description"].ToString();
            _modified = false;
        }
        UpdateCanvas(sender, e);
    }
}
```

Obsługuje wybieranie operacji pobranych z bazy danych.

Metoda RefreshUnitermsList:

```
private void RefreshUnitermsList()
{
    uiUnitermsList.Items.Clear();

    foreach (DataRow dataRow in _database.Query("select id,name from uniterms;"))
    {
        uiUnitermsList.Items.Add(dataRow["id"]+" "+dataRow["name"]);
    }
}
```

Pobiera operacje z bazy danych i wstawia je na listę wyboru.

Klasa Database:

```
class Database
{
    private static String _connectionString = "Data Source=MSI;Initial Catalog=MASI;User
ID=sa;Password=sa";
    private SqlConnection _connection;

    public Database()
    {
        _connection = new SqlConnection(_connectionString);
    }
    public void Connect()
    {
        try
        {
            _connection.Open();
        }
        catch (InvalidCastException ex)
        {
            throw ex;
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }

    public void Disconnect()
    {
        try
        {
            _connection.Close();
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
    public DataRowCollection Query(string query)
    {
        DataTable tab = new DataTable();

        this.Connect();

        if (_connection.State == ConnectionState.Open)
        {
            SqlDataAdapter ad = new SqlDataAdapter(query, _connection);

            ad.Fill(tab);
        }
        else
        {
            MessageBox.Show("Nie można połączyć się z bazą danych");
        }

        this.Disconnect();

        return tab.Rows;
    }
}
```

```
public DataRowCollection Execute(string query)
{
    DataTable tab = new DataTable();

    this.Connect();

    if (_connection.State == ConnectionState.Open)
    {
        SqlCommand cmd = new SqlCommand(query, _connection);

        cmd.ExecuteNonQuery();
    }
    else
    {
        MessageBox.Show("Nie można połączyć się z bazą danych");
    }

    this.Disconnect();

    return tab.Rows;
}
```

Posiada metody do obsługi połączenia z bazą danych, wykonania zapytania i polecenia.

W klasie Drawing najważniejszą metodą jest metoda Update:

```
public void Update()
{
    using (DrawingContext dc = Visual.RenderOpen())
    {
        ClearDrawingContext(dc);

        //double margin = FontSize/3;
        double margin = 50;
        double origin = margin + FontSize;
        double radiusX = FontSize * 2 / 3;
        double radiusY = (FontSize + 2) * 3 / 4;
        Pen pen = new Pen(Brushes.Black, FontSize / 6);
        string textCycle = CycleVariable + " ";

        if (CycleVariable.Length > 0 && CycleOperation.Length > 0)
        {
            dc.DrawEllipse(Brushes.White, pen, new Point(origin, origin), radiusX, radiusY);
            dc.DrawLine(pen, new Point(origin + radiusX, origin - radiusY), new Point(origin - radiusX,
origin + radiusY));
            if (!Exchanged)
            {
                textCycle += CycleOperation;
            }
            dc.DrawText(GetFormattedText(textCycle), new Point(origin + radiusX * 2, origin -
GetFormattedText(textCycle).Height / 2));
        }

        if (EliminationA.Length > 0 && EliminationB.Length > 0 && EliminationCondition.Length > 0 &&
EliminationOperation.Length > 0)
        {
            double originX = origin - radiusX;
            double originY = origin + radiusY * 4;
            double verticalLineSize = FontSize / 3;
            double innerMargin = verticalLineSize * 2;
            string textElimination = EliminationA + EliminationOperation + " " + EliminationB +
EliminationOperation + " " + EliminationCondition;
            if (Exchanged)
            {
                originX = origin + radiusX * 3 + GetFormattedText(textCycle).Width;
                //originY = origin - verticalLineSize*2 - GetFormattedText(textElimination).Height / 4;
                originY = origin - GetFormattedText(textCycle).Height / 2;
                textElimination += " _ " + CycleVariable;
            }
            textElimination += " - ?";
            FormattedText formattedTextElimination = GetFormattedText(textElimination);
            dc.DrawLine(pen, new Point(originX, originY - formattedTextElimination.Height / 4), new
Point(originX + formattedTextElimination.Width + innerMargin * 2, originY - formattedTextElimination.Height / 4));
            dc.DrawLine(pen, new Point(originX, originY - formattedTextElimination.Height / 4 -
verticalLineSize), new Point(originX, originY - formattedTextElimination.Height / 4 + verticalLineSize));
            dc.DrawLine(pen, new Point(originX + formattedTextElimination.Width + innerMargin * 2, originY -
formattedTextElimination.Height / 4 - verticalLineSize), new Point(originX + formattedTextElimination.Width +
innerMargin * 2, originY - formattedTextElimination.Height / 4 + verticalLineSize));
            dc.DrawText(formattedTextElimination, new Point(originX + innerMargin, originY));
        }

        dc.Close();
    }
}
```

Jest ona odpowiedzialna za rysowanie unitermów.

Przykład:

MainWindow

9 uniterm 1
10 następny uniterm

$\emptyset \times P(f_x)$

$\overline{\hspace{1cm}}$
R; S; u - ?

Nazwa: uniterm 1 Opis: przykładowy uniterm

Nowy
Edytuj cykl. równoleg.
Edytuj eliminowanie
Czcionka: Arial
Rozmiar: 28
☐ Zamień
Zapisz
Usuń

Po zamianie:

MainWindow

9 uniterm 1
10 następny uniterm

$\emptyset \times \overline{\hspace{1cm}}$
R; S; u_x - ?

Nazwa: uniterm 1 Opis: przykładowy uniterm

Nowy
Edytuj cykl. równoleg.
Edytuj eliminowanie
Czcionka: Arial
Rozmiar: 28
☒ Zamień
Zapisz
Usuń

4. Wnioski.

Projekt wymagał i był pomocny w zapoznaniu się z algebrą algorytmów, narzędziami Visual Studio, technologią WPF i MSSQL, a także z językiem C#. Wszystkie założenia projektowe udało się zrealizować.