POLITECHNIKA POZNAŃSKA

Oprogramowanie dla telefonów z systemem Windows Phone 8 obsługujących technologię NFC do sprawdzania listy obecności.

Programowanie Kart Elektronicznych
Systemy Informatyczne w Zarządzaniu
Michał Majer 94447

WSTĘPNY OPIS PROJEKTU

Niniejszy dokument został przygotowany w celach zaliczenia przedmiotu „Programowanie Kart Elektronicznych” i jest dokumentacją projektu aplikacji o roboczej nazwie „ELS Reader”.

Oprogramowanie będzie miało na celu automatyzację weryfikacji obecności studentów na zajęciach. Zainstalowane na wybranym urządzeniu oprogramowanie będzie odnotowywać obecność uczestników na zajęciach poprzez zbliżenie ELS do urządzenia. Po zbliżeniu legitymacji do telefonu z aktywnym oprogramowaniem na ekranie zostaną zaprezentowane informacje o posiadaczu karty tym samym potwierdzając obecność danego studenta na bieżących zajęciach. Zebrane informacje zostaną przechowane w wewnętrznej bazie danych aplikacji, a następnie wyeksportowane do usługi SkyDrive.

ROZDZIAŁ 1. INFORMACJE TECHNICZNE

1.1. Wymagania

Do poprawnego działania aplikacji wymagane jest posiadanie telefonu z zainstalowanym systemem operacyjny Windows Phone w wersji 8 obsługującego technologię NFC.

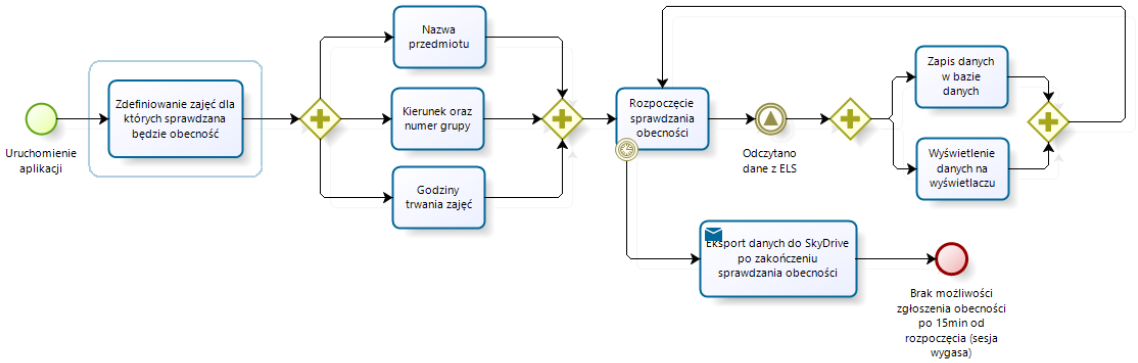
1.2. Środowisko i języki programowania

Do realizacji projektu zostało wykorzystane środowisko deweloperskie **Visual Studio 2013** oraz **Blend** dla Visual Studio 2013. Aplikacja została napisana w języku Windows Phone App **C#** (Visual Studio), a wizualizacja aplikacji na urządzeniu została zinterpretowana za pomocą znaczników **XAML** (wsparcie przy pomocy Microsoft Blend).

ROZDZIAŁ 2. SZCZEGÓŁOWY OPIS OPROGRAMOWANIA

2.1. Proces aplikacji

Cykl życia aplikacji został zamodelowany i przedstawiony na rysunku 1.



Rysunek 1. Cykl życia aplikacji

W momencie uruchomienia aplikacji przez użytkownika – rozpoczynany jest cykl życia projektu. Wykładowca definiuje zajęcia na których chce przeprowadzić weryfikację obecności uczestników: wprowadza nazwę przedmiotu, kierunek grupy, numer grupy oraz godziny trwania zajęć. Po zatwierdzeniu wpisanych danych urządzenie przechodzi w tryb sprawdzania obecności i oczekuje na zbliżenie legitymacji studenckiej. Poprawny odczyt danych z przybliżonej karty powoduje zapis przekazanych danych i wyświetlenie ich na ekranie urządzenia. Urządzenie będzie oczekiwało na zbliżenie ELS do 15 min od rozpoczęcia zajęć. Po tym czasie sesja wygaśnie, a zebrane dane zostaną wyeksportowane do usługi SkyDrive.

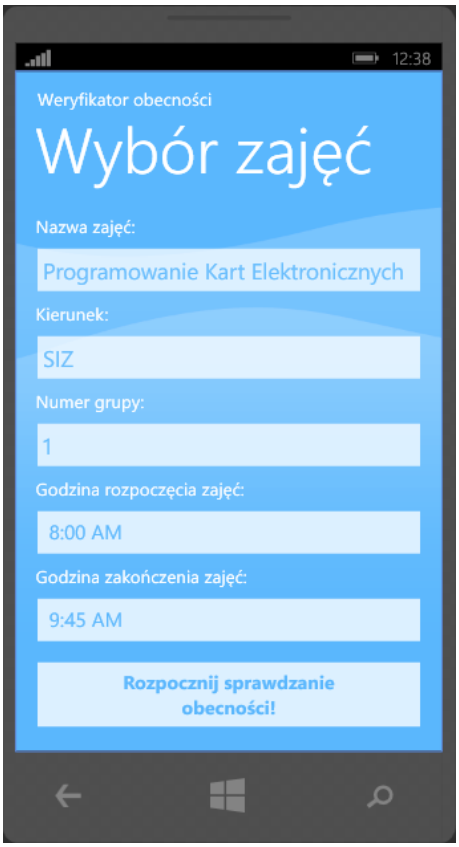
2.2. Interfejs i obsługa aplikacji

W celu zapewnienia jak najłatwiejszej obsługi całego procesu aplikacji wykonany został przejrzysty i łatwy w obsłudze interfejs. Wraz z uruchomieniem aplikacji użytkownik zobaczy interfejs widoczny na rysunku nr 2. Widok programu umożliwia użytkownikowi zdefiniowanie zajęć.

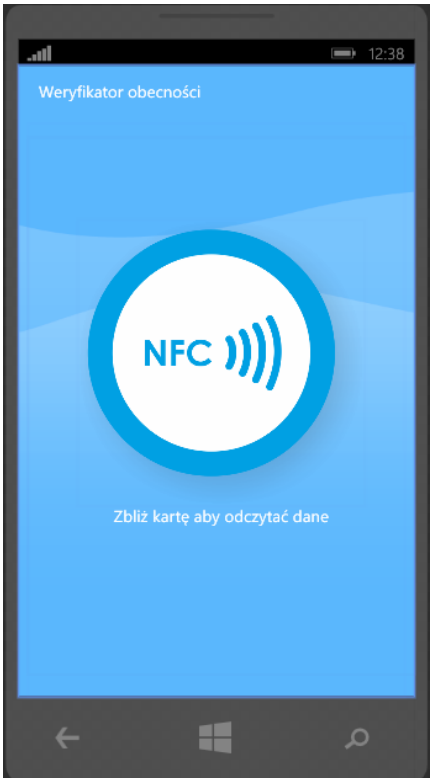
W widoczne pola należy wpisać:

- Nazwę prowadzonych zajęć
- Kierunek grupy
- Numer grupy
- Godzinę rozpoczęcia zajęć
- Godzinę zakończenia zajęć

Po wybraniu przycisku „Rozpocznij sprawdzanie obecności!” urządzenie rozpocznie oczekiwanie na zbliżenie elektronicznej legitymacji studenckiej.



Rysunek 2. Ekran startowy - definiowania zajęć



Rysunek 3. Zachęcenie do zbliżenia karty

Zrzut ekranu przestawiony na rysunku 3 zachęca do zbliżenia karty w celu potwierdzenia swojej obecności na zajęciach. W momencie zbliżenia elektronicznej karty studenckiej algorytm odczytuje dane i weryfikuje czy dane są zgodne z oczekującymi. W przypadku gdy dane nie są odpowiednie aplikacja nie zareaguje i będzie oczekiwała na kolejne „zbliżenie". Gdy pozyskane dane będą prawidłowe aplikacja przejdzie do widoku potwierdzenia (rysunek 4).

Rysunek 4.Pomyślny odczyt danych

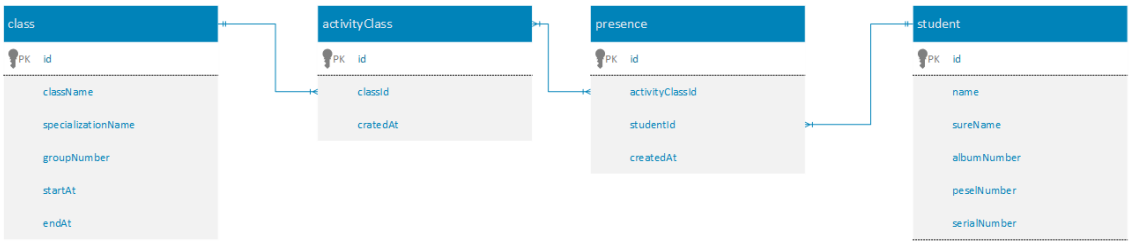
Po zakończeniu sesji sprawdzania obecności (15 min tzw. „studencki kwadrans") aplikacja eksportuje zebrane dane do usługi SkyDrive.

Przykładowy kod definiujący interfejs:

```
<phone:PhoneApplicationPage
    x:Class="ELSReader.PageMoveCloserELS"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone"
    xmlns:shell="clr-namespace:Microsoft.Phone.Shell;assembly=Microsoft.Phone"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    FontFamily="{StaticResource PhoneFontFamilyNormal}"
    FontSize="{StaticResource PhoneFontSizeNormal}"
    Foreground="{StaticResource PhoneForegroundBrush}"
    SupportedOrientations="Portrait" Orientation="Portrait"
    mc:Ignorable="d"
    shell:SystemTray.IsVisible="True">

    <!--LayoutRoot is the root grid where all page content is placed-->
    <Grid x:Name="LayoutRoot">
        <Grid.Background>
            <ImageBrush Stretch="None" ImageSource="Assets/bg.jpg"/>
        </Grid.Background>
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="*"/>
        </Grid.RowDefinitions>

        <StackPanel x:Name="TitlePanel" Grid.Row="0" Margin="12,17,0,28">
            <TextBlock Text="Weryfikator obecności" Style="{StaticResource
PhoneTextNormalStyle}" Margin="12,0"/>
        </StackPanel>

        <Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0"/>

        <StackPanel x:Name="ContentStackPanel" Grid.Row="1" Margin="12,17,0,28">
            <Image Height="346" Source="/Assets/nfc.png" VerticalAlignment="Center"
HorizontalAlignment="Center" Margin="61,100,61,0"/>
            <TextBlock TextWrapping="Wrap" Text="Zbliż kartę aby odczytać dane"
HorizontalAlignment="Center"/>
        </StackPanel>

    </Grid>

</phone:PhoneApplicationPage>
```

2.3. Baza danych

Aplikacja wykorzystuje bazę danych SQLite do przechowywania danych. Dane zapisywane są w tabelach przedstawionych na rysunku 5.



Rysunek 5. Diagram bazy danych

Tabele są odzwierciedleniem utworzonych klas w programie. Tabel `student` przechowuje dane odczytane z legitymacji studenckiej, zapisuje dane takie jak: imię, nazwisko, numer albumu, numer pesel oraz numer seryjny legitymacji. Tabela `class` odnosi się do utworzonych przez operatora aplikacji zajęć. Przechowywanie tych danych w tabeli ułatwi wykładowcy proces wyboru zajęć dla których weryfikowana będzie obecność. Tabele `activityClass` oraz `presence` dotyczą procesu odnotowywania obecności studentów dla utworzonych zajęć.

Kod definicji tabel:

```
public sealed class Student
{
    [PrimaryKey, AutoIncrement]
    public int id { get; set; }
    public string name { get; set; }
    public string sureName { get; set; }
    public int albumNumber { get; set; }
    public int peselNumber { get; set; }
    public int serialNumber { get; set; }

    public override string ToString()
    {
        return name + " " + sureName;
    }
}

public sealed class Class
{
    [PrimaryKey, AutoIncrement]
    public int id { get; set; }
    public string className { get; set; }
    public string specializationName { get; set; }
    public int groupNumber { get; set; }
    public DateTime startAt { get; set; }
    public DateTime EndAt { get; set; }

    public override string ToString()
    {
        return className;
    }
}
```

2.4. Odczyt danych wykorzystując technologię NFC

NFC (Near Field Communication) – jest to komunikacja bezprzewodowa niewielkiego zasięgu. Umożwia ona przenoszenie informacji między telefonem, a innym urządzeniem NFC (telefon, tag, urządzenia płatnicze).

Ograniczenia w Windows Phone 8 dotyczące NFC:

- Brak możliwości zabezpieczenia przed zapisem tagów.
- Brak możliwości sformatowania tagów do NDEF. Aby działały z naszym telefonem należy kupić sformatowane tagi lub użyć urządzania wspierającego formatowanie (np. Android). W przypadku uszkodzenia taga też nie jesteśmy w stanie go naprawić.
- Tagi mogą zawierać tylko wiadomość w formacie NDEF.
- Brak możliwości użycia całej dostępnej pamięci taga – znaczniki, korekcja błędów, …
- API nie wspiera zapisywania wiadomości w formacie NDEF domyślnie.
- Możemy to zrobić przygotowując wiadomość w formacie binarnym lub
- posłużyć się zewnętrzną biblioteką.
- Urządzenie domyślnie nie jest w stanie odczytać wszystkich wiadomości zapisanych na tagu. System wspiera odczyt tylko pierwszej wiadomości.
- Dodatkowe mogą być odczytane za pomocą własnej aplikacji.
- Nie ma możliwości odczytania taga przez aplikację działającą w tle.

Z powodu ograniczeń WP8 dotyczących NFC realizacja projektu (a konkretniej odczyt danych z ELS) jest niemożliwy na dzień dzisiejszy!

Przykładowy kod programu do obsługi NFC w Windows Phone 8:

1. Weryfikacja czy urządzenie posiada odbiornik NFC:

```
ProximityDevice proximityDevice = ProximityDevice.GetDefault();
if (proximityDevice != null)
{
    // Urządzenie ma moduł NFC
    proximityDevice.DeviceArrived += proximityDevice_DeviceArrived;
    proximityDevice.DeviceDeparted += proximityDevice_DeviceDeparted;
}
else
{
    // Brak modułu NFC
}
```

2. Odczyt tagu:

```
ProximityDevice proximityDevice = ProximityDevice.GetDefault();
if (proximityDevice != null)
{
 long subscribedMessageId
 = proximityDevice.SubscribeForMessage("WriteableTag", Handler);
}

void Handler(ProximityDevice device, ProximityMessage message)
{
```

```
 int writableSize = BitConverter.ToInt32(message.Data.ToArray(), 0);
 this.ShowMessage("MessageType: " + message.MessageType + "\n"
 + "DataAsString:" + message.DataAsString + "\n"
 + "SubscriptionId: " + message.SubscriptionId + "\n"
 + "Size of tag: " + writableSize + "\n");
}

proximityDevice.StopSubscribingForMessage(subscribedMessageId);
```

3. Odczyt tagu(2):

```
this.subscribedMessageId = this.proximityDevice.SubscribeForMessage(
 "Windows.mySubType", Handler2);

private void Handler2(ProximityDevice device,
 ProximityMessage message)
{
 this.proximityDevice.StopSubscribingForMessage(
 this.subscribedMessageId);

 byte[] buf = message.Data.ToArray();
 this.ShowMessage("MessageType: " + message.MessageType + "\n"
 +"Data:"+Encoding.Unicode.GetString(buf, 0, buf.Length)+"\n"
 + "SubscriptionId: " + message.SubscriptionId + "\n");
}
```

4. Zapis danych do tagu:

```
this.proximityDevice.PublishMessage("Windows.SampleMessage", "Hello
World",
 Handler);

this.proximityDevice.PublishUriMessage(new
System.Uri("http://www.onet.pl",
 Handler);

this.proximityDevice.PublishBinaryMessage("Windows:WriteTag.mySubType",
 Encoding.Unicode.GetBytes("Hello World").AsBuffer(), Handler);

Uri uri = new Uri("http://www.wp.pl");
byte[] buffer = Encoding.Unicode.GetBytes(uri.ToString());
proximityDevice.PublishBinaryMessage("WindowsUri:WriteTag",
 buffer.AsBuffer(), Handler);

private void Handler(ProximityDevice sender, long messageId)
{
 this.proximityDevice.StopPublishingMessage(messageId);
 this.ShowMessage("Wysłano / Zapisano!");
}
```

2.5. Usługa SkyDrive

Windows Live Skydrive jest wydzieloną przestrzenią na serwerach Microsoftu, na której umieszczać możemy własne pliki i dane. Usługa zadebiutowała w Polsce w maju 2008 roku. Obecnie Windows Live Skydrive zapewnia aż 25 GB pojemności do wykorzystania i to w zupełności za darmo! Zalet z tego faktu jest mnóstwo, a przede wszystkim jest to idealny sposób na uzyskiwanie dostępu do plików niezależnie od komputera, poprzez który łączymy się z siecią. W projekcie usługa zostanie użyta do przechowywania danych pozyskanych przez aplikację. Umożliwi to łatwy dostęp i możliwość prostej edycji przekazanych informacji.

Przykładowy kod programu:

1. Logowanie do systemu SkyDrive

```
private void skydrive_SessionChanged(object sender, LiveConnectSessionChangedEventArgs
e)
{

            if (e != null && e.Status == LiveConnectSessionStatus.Connected)
            {
                this.client = new LiveConnectClient(e.Session);
                this.GetAccountInformations();
            }
            else
            {
                this.client = null;
                InfoText.Text = e.Error != null ? e.Error.ToString() : string.Empty;
            }

}

 private async void GetAccountInformations()
 {
            try
            {
                LiveOperationResult operationResult = await
this.client.GetAsync("me");
                var jsonResult = operationResult.Result as dynamic;
                string firstName = jsonResult.first_name ?? string.Empty;
                string lastName = jsonResult.last_name ?? string.Empty;
                InfoText.Text = "Welcome " + firstName + " " + lastName;
            }
            catch (Exception e)
            {
                InfoText.Text = e.ToString();
            }

 }
```

2. Eksport danych

```
private string fileName = "sample.dat";
private IsolatedStorageFile isf = IsolatedStorageFile.GetUserStoreForApplication();

private void CreateFileIntoIsolatedStorage()
{
            if (isf.FileExists(fileName))
            {
                isf.DeleteFile(fileName);
            }

            IsolatedStorageFileStream isfStream =   new
IsolatedStorageFileStream(fileName,  FileMode.Create,
IsolatedStorageFile.GetUserStoreForApplication());

            byte[] output = new byte[25];

            for (int i = 0; i < 25; i++)
            {
                output[i] = (byte)(i);
            }

            isfStream.Write(output, 0, output.Length);

            isfStream.Close();

}
```

ROZDZIAŁ 3. WNIOSKI

Pomimo wsparcia NFC dla urządzeń z systemem Windows Phone w wersji 8 nie jest możliwe odczytanie danych z elektronicznej legitymacji studenckiej. Biblioteki dostarczone przez producenta umożliwiają odczyt danych tylko i wyłącznie w formacie NDEF. Liczne ograniczenia nałożone przez WP dla technologii NFC uniemożliwiają realizację tematu dla wybranej platformy.

Rozwiązaniem problemu pozostaje zmiana platformy na konkurencyjną platformę Android.

LITERATURA

[1] Ian Griffiths „Programowanie C# 5.0”

[2] Daniel Vaughan „Windows Phone Unleashed”

[3] Kyle Byrns „Beginning Windows 8 Application Development XAML Edition”