Programowanie Zespołowe 2016

Bluetooth Messenger

Wojciech Polak, Klaudia Głocka, Rafał Ziembiński, Konrad Chojnecki

24 listopada 2016

Spis treści

1 Opis aplikacji

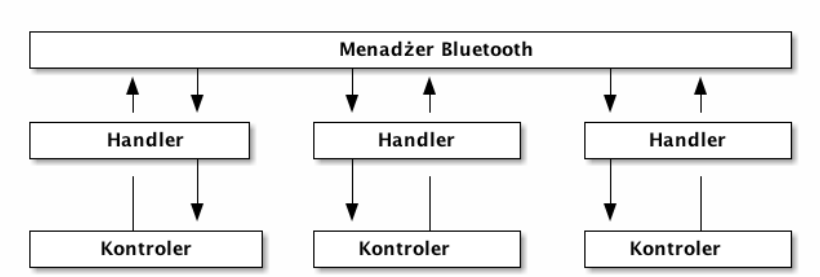
Aplikacja ma za zadanie połączenie dwóch lub więcej urządzeń, poprzez protokół komunikacyjny *Bluetooth*, w celu zapewnienia podstawowej komunikacji w postaci wiadomości tekstowych.

1.1 Użyte technologie

Do stworzenia aplikacji użyto zestawu narzędzi firmy *Xamarin*. Kod źródłowy aplikacji został napisany głównie w języku C#. Użyte biblioteki i oprogramowanie pozwoliło współdzielić kod dla wielu platform, tym samym umożliwiając szybki rozwój aplikacji zarówno na system **iOS** jak i **Android**.

1.2 Zarys architektury aplikacji

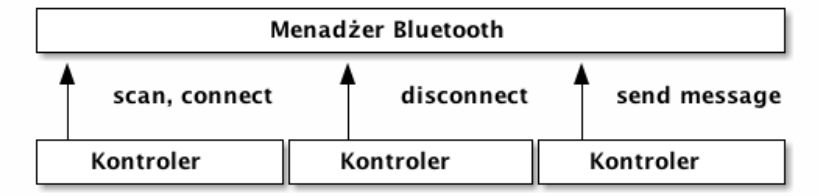
Aplikacja została stworzona w oparciu o wzorzec projektowy *Model - View - Controller*.



1.2.1 Moduł Menadżera Bluetooth

Menadżer Bluetooth odpowiada za najniższą warstwę aplikacji. Każdy z systemów - odpowiednio **iOS** jak i **Android** posiadają inne Programistyczne Interfejsy Aplikacji (w skrócie *API*). Najniższa warstwa będąca *najbliżej sprzętu* została rozdzielona pomiędzy systemy. Dlatego ten menadżer został zaimplementowany jako dwa moduły.

Ponieważ wymagane jest aby aplikacja w przyszłości była łatwo rozszerzalna, została wprowadzona warstwa abstrakcji - `IBluetoothManager` która służy jako punkt wyjścia i wejścia dla innych modułów.



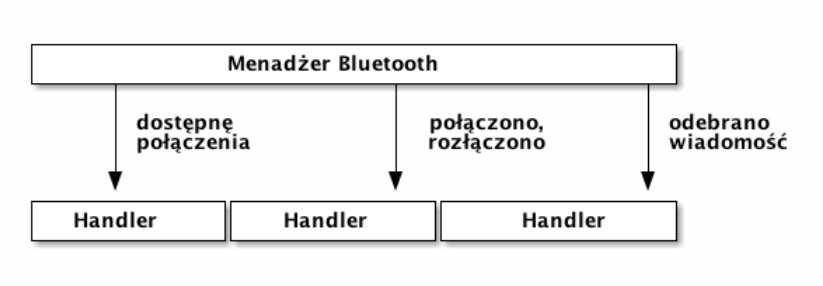
1.2.2 Asynchroniczność aplikacji

Aplikacja jest zależna od zasobów zewnętrznych takich jak sieć *Bluetooth*. W momencie gdy system czeka na nawiązanie połączenia z drugą komórką jest wymagane aby użytkownik cały czas miał aplikację interaktywną i responsywną.

Aby aplikacja nie blokowała interfejsu użytkownika wszystkie akcje wykonywane przez warstwy niższe i pośrednie muszą być **asynchroniczne**. Rozwiązaniem są moduły `Handlera` których zadaniem jest reakcja na wydarzenia.

1.2.3 Moduł `Handler`

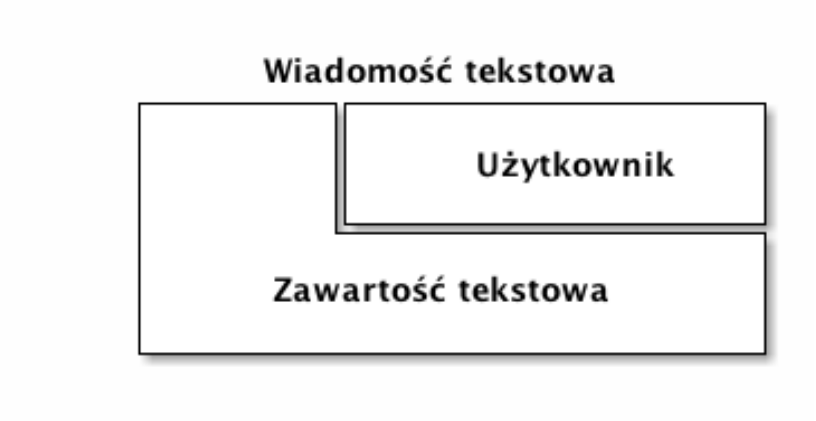
W momencie gdy zadanie zostanie wykonane (np. użytkownik zostanie połączony z innym), wiadomość zostaje wysłana do odpowiedniego `Handler` który reaguje w odpowiedni sposób do danych wejściowych.



W projekcie znajdują się dwa moduły typu `Handler`

1. Moduł `Message Handler` `Message Handler` odpowiada za łączenie z innymi użytkownikami. Do zadań tego modułu należą:
   - Reakcja na pobraną listę dostępnych w pobliżu użytkowników - pseudonimy użytkowników są wyświetlane na ekranie.
   - Reakcja na połączenie się z danym użytkownikiem - następuje zmiana widoku na widok wysłanych i odebranych wiadomości.
2. Moduł `Connection Handler` `Connection Handler` odpowiada za reakcję na wiadomości odebrane od innego użytkownika. Wiadomości takie są wyświetlane w czytelnej formie na ekranie telefonu.

1.2.4 Modele



1. Użytkownik Aplikacja przechowuje informacje o użytkowniku takie jak:
   - pseudonim
   - unikalny identyfikator oparty o technologię `GUID4`
2. Wiadomość Wysyłane i odbierane wiadomości mają format:
   - Użytkownik
   - Wiadomość tekstowa

1.2.5 Kontrolery

Kontrolery odpowiadają za zarządzanie danymi które zostały odebrane przez moduły `Handler`. Często wymagane jest aby dane te zostały odpowiednio spreparowane zanim zostaną wyświetlone na ekranie. Dobrą praktyką jest, aby w dalszych widokach **nie było żadnej logiki biznesowej**. Dlatego każda operacja na danych musi się odbyć w kontrolerze.

Kontrolery są modułami które odbierają wydarzenia (np. naciśnięcie przycisku, wpisanie tekstu, gesty czy ruch zarejestrowany przez akcelerator) które zostały wykonane w odpowiednich widokach. Kontrolery reagują wydarzenia i na podstawie zawartości wydarzeń przesyłają odpowiednie komendy do pozostałych modułów, najczęściej do Menadżera Bluetooth.

1.2.6 Widoki

Aplikacja składa się z dwóch widoków.

1. Widok z możliwymi połączeniami. W tym widoku użytkownik może zobaczyć wszystkich innych użytkowników, którzy są w zasięgu. Urządzenie skanuje obszar w określonym interwale czasowym. Użytkownik może nawiązać bezpośrednie połączenie z jednym użytkownikiem tym samym przechodzą do widoku drugiego.
2. Widok wymiany wiadomości. W tym widoku użytkownik wysyła i odbiera wiadomości nadane przez drugiego użytkownika. Na raz możliwa jest rozmowa tylko z jednym użytkownikiem. Użytkownik oprócz wysyłania wiadomości może także zakończyć rozmowę tym samym wracając do widoku pierwszego.

2 Podział prac

2.1 Rafał Ziembiński - moduł `BluetoothManager` dla systemu Android

2.2 Klaudia Głocka - moduł `ConnectionHandler`

2.3 Konrad Chojnecki - moduł `MessageHandler`

2.4 Wojciech Polak - moduł `BluetoothManager` dla systemu iOS

3 Stan prac

3.1 Na dzień 01-11-2016

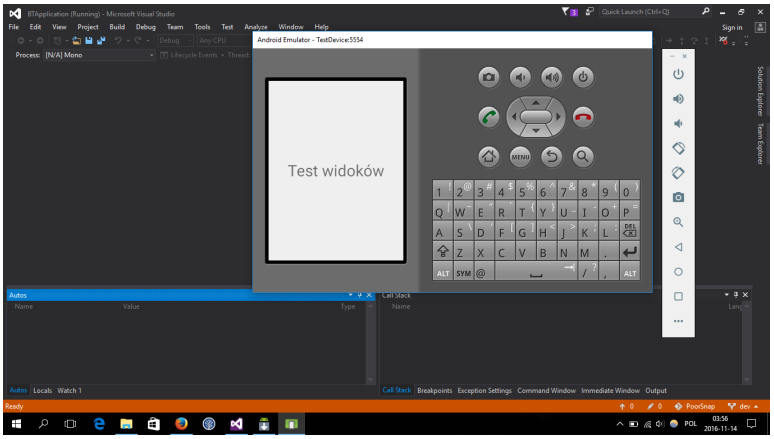
1. Przerwa w pracy w wyniku dni wolnych od pracy
2. Poprawa dokumentu opisującego projekt. Wykorzystanie w tym celu LaTeX.
3. Konfiguracja środowisk programistycznych:
    - Próby instalacji IDE, wymaganych bibliotek i narzędzi pracy
    - Konfiguracja maszyn wirtualnych oraz urządzeń natywnych

3.2 Na dzień 10-11-2016

1. Reinstalacja systemu operacyjnego Microsoft Windows na jednym stanowisku pracy, konfiguracja wszystkich potrzebnych bibliotek, narzędzi i edytorów.
2. Usunięcie Visual Studio 2013 na drugim stanowisku pracy. Konfiguracja Visual Studio 2015.
3. Aktualizacja dokumentacji o grafy i wykresy połączeń pomiędzy modułami

3.3 Na dzień 17-11-2016

- **Rafał Ziembiński**: Reinstalacja całego systemu operacyjnego umożliwiła uruchomienie emulatora Android wraz z działającym prototypem aplikacji.



Rozpoczęty research w sprawie Bluetooth na Android.

- **Klaudia Głocka**: Rozpoczęty research i pierwsze testy związane z obsługą Visual Studio. Rozpoczęty kurs języka C#. Pierwsze próby z systemem kontroli wersji Git.
- **Konrad Chojnecki**: Research dotyczący architektury zestawu narzędzi Xamarin. Research dotyczący widoków aplikacji.
- **Wojciech Polak**: Dalszy research w sprawie Bluetooth na iOS: Głównie oparty o `https://developer.xamarin.com/api/namespace/CoreBluetooth/` API `CoreBluetooth` pozwala na pracę z `Bluetooth Low Energy`. Wciąż nie jest pewne, czy będzie możliwa bezproblemowa komunikacja między iOS a Android. Znalezione zostało rozwiązanie pośrednie - gotowy moduł działający zarówno dla iOS jak i dla Android - `https://components.xamarin.com/view/ch.arendi.blelibrary`

3.4 Na dzień 24-11-2016

- **Rafał Ziembiński** Pobieranie emulatorów różnych urządzeń Android w celu sprawdzenia integralności aplikacji na różnych platformach i wersjach systemu. Badanie przestrzeni nazw `Android.Bluetooth`. Sprawdzenie modułu działającego zarówno na iOS jak i dla Android.
- **Klauda Głocka** Przygotowanie widoków aplikacji oraz zdobywanie wiedzy jak je podpiąć pod moduł `MessageHandler`.
- **Wojciech Polak** Znalezione `API ExternalAccessory` pozwalające urządzeniom iOS na interakcje z urządzeniami Bluetooth. [`https://developer.xamarin.com/api/namespace/ExternalAccessory/`] Aktualizacja środowiska Xamarin Studio z powodów wymuszonej aktualizacji `IDE` XCode 8. Walka z błędami przy wdrażaniu na natywne urządzenie.