

POLITECHNIKA ŚLĄSKA W GLIWICACH
WYDZIAŁ INŻYNIERII BIOMEDYCZNEJ

Sprawozdanie

Telemedycyna i technologie sieciowe
System rozpoznawania mówcy

Mario Bas, Bartosz Klimek

Gliwice, 26 stycznia 2019

1. Wstęp i analiza zagadnienia

W tym rozdziale opisany zostanie cel projektu oraz technologie które zostały wykorzystane w celu zaimplementowania rozwiązania.

1.1 Cel projektu

Głównym celem projektu było zapoznanie się istniejącymi metodami bezstratnej transmisji oraz nagrywania ludzkiej mowy, a następnie stworzenie aplikacji umożliwiającej transmisję nagranej mowy na serwer. Projekt jest częścią większej pracy mającej na celu stworzeniu systemu identyfikacji mówcy. Analizą i przetworzeniem nagranej mowy zajmuje się serwer na który wysyłany jest dźwięk, jednakże opis działania serwera, wyłączając część związaną z transmisją danych, znajduje się w innej części sprawozdania.

1.2 Analiza obiektowa projektowanego rozwiązania

W celu nagrania mowy stworzono prosty interfejs graficzny wykorzystując technologię Windows Forms oraz bibliotekę NAudio. Dane dźwiękowe zostały wysyłane na serwer w postaci strumienia bajtów. Dane na serwerze zostają przetworzone, co jest zadaniem algorytmu stworzonego w ramach innego projektu, a następnie w postaci bajtów serwer odsyła do klienta decyzję o udzieleniu dostępu. Aby zapewnić bezstratną oraz szybką transmisję danych skorzystano z modelu TCP/IP, przy czym w projekcie skupiono się głównie na warstwie transportowej modelu. Strona kliencka została stworzona w środowisku *Visual Studio .NET*, a strona serwerowa w środowisku *MatLab*. Aktualnie nie zakłada się działania wielowątkowego dla serwera, podczas połączenia serwera z jednym klientem, próba połączenia kolejnego będzie przerywana.

Bezpieczeństwo danych zapewniono wykorzystując wbudowane w środowisko *Visual Studio .NET* biblioteki szyfrowania symetrycznego AES Rijndael. Klucz oraz wektor inicjujący nie są przesyłane pomiędzy klientem i serwerem. Założone, że w ramach projektu będą one wczytywane bezpośrednio z plików tekstowych po stronie klienta jak i serwera.

2. Specyfikacja wewnętrzna

Projekt złożony jest z dwóch głównych warstw czyli: warstwy serwerowej stworzonej w języku Matlab oraz warstwy klienckiej stworzonej w języku C#. Dodatkowo stworzona została biblioteka w języku C#, na rzecz wykorzystania algorytmów szyfrowania symetrycznego AES Rijandel w środowisku Matlab. Specyfikacja poszczególnych warstw znajduje się poniżej:

2.1 Warstwa kliencka

Ta część projektu jest zaimplementowaną logiką schowaną pod prostym interfejsem graficznym, stworzonym w technologii *WinForms*. Wszystkie operacje zawarte są w jednej klasie o nazwie `DataWaveOperations()`, która zawiera poniższe metody:

1.

```
public void StartRecording ()
```

funkcja odpowiedzialna za nagranie głosu za pomocą mikrofonu oraz zapisanie nagrania na dysku w formacie .wav. Funkcja jest typu void czyli nie zwraca żadnej zmiennej oraz jest bezargumentowa.

2.

```
public void StopRecording ()
```

metoda typu void oraz również bezargumentowa odpowiedzialna za zakończenie nagrywania głosu.

3.

```
public byte [] OpenWaveFile ()
```

bezargumentowa funkcja odpowiedzialna za wczytanie nagrania w bajtowej reprezentacji z dysku,

- Zwracane: tablica bajtów odpowiadająca próbkom nagrania.

4.

```
public void PlayWaveFile ()
```

funkcja typu void zajmująca się odtworzeniem powstałego nagrania,

- Argumenty: obiekt klasy *WaveOutEvent* oraz obiekt klasy *WaveFileReader*

5.

```
public string SendDataAndReceiveAnswer()
```

bezargumentowa metoda odpowiedzialna za wysłanie danych za pomocą protokołu TCP/IP na serwer oraz odebranie od niego odpowiedzi

- Zwracane: string z odpowiedzią serwera.

6.

```
public static byte[] CutWavInfo()
```

działanie tej funkcji ogranicza się do wycięcia początkowych bajtów odczytanego nagrania w formacie .wav, które niosą informację na temat samego nagrania

- Argumenty: tablica bajtów z odczytanym sygnałem
- Zwracane: tablica bajtów z usuniętym nagłówkiem informacyjnym.

7.

```
private static string GetAnswer()
```

zadaniem tej funkcji było konwersja bajtowego formatu odpowiedzi na czytelny napis,

- Argumenty: tablica bajtów odpowiedzi, liczba bajtów do konwersji
- Zwracane: string z odkodowaną odpowiedzią.

8.

```
public static byte[] CreateDataPacket()
```

z kolei ta funkcja była odpowiedzialna za otworzenie strumienia bajtów niezbędnego do strumieniowego przesyłu bajtów,

- Argumenty: tablica bajtów nagrania
- Zwracane: strumień bajtów.

2.2 Warstwa serwerowa

Serwer odbierający, przetwarzający oraz wysyłający odpowiedź stworzony został w środowisku Matlab za pomocą jednego skryptu. Działanie serwera jest oparte na nieskończonej pętli `while()`, a program ciągle oczekuje na połączenie klienta. Wewnątrz pętli wykorzystywane są funkcje głównie związane z obsługą protokołu TCP/IP:

1. `tcpip()`

jest to funkcja inicjalizująca obiekt TCP/IP, w tym wypadku serwer,

- Argumenty: `host`, `port`, argument `'NetworkRole' 'Server'`, który wskazuje, że obiekt `tcpip` będzie serwerem
- Zwracane: obiekt typu `tcpip`.

2. `fopen()`

jedna z 4 funkcji, wykonująca operacje na strumieniach - w tym wypadku otwarcie strumienia. Niezwracająca żadnej zmiennej.

- Argument: obiekt `tcpip`.

3. `fread()`

kolejna funkcja obsługująca strumień,

- Argumenty: obiekt `tcpip`, liczba bajtów do przyjęcia, typ odbieranych danych
- Zwracane: tablica danych w tym wypadku bajtów

4. `fwrite()`

następna z kolei funkcja zajmująca się strumieniami, odpowiedzialna za wysłanie ciągu bajtów tym samym połączeniem `tcpip`,

- Argumenty: obiekt `tcpip`, tablica bajtów oraz string określający typ wysyłanych danych, funkcja ta nie zwraca żadnej zmiennej.

5. `fclose()`

funkcja typu `void` zamykająca strumień danych, w argumencie pobiera obiekt typu `tcpip`.

2.3 Biblioteka AesLib

Na potrzeby projektu została stworzona samodzielnie biblioteka w języku C# o nazwie `AesLib.dll`, która ma za zadanie udostępnienie metod szyfrowania i deszyfrowania danych za pomocą symetrycznego algorytmu AES Rijandel. Zgodnie z powyższym w tej bibliotece została umieszczona klasa *Crypto*, która ma zaimplementowane dwie metody:

1.

```
public static byte [] EncryptBytes()
```

funkcja ta, jak sama nazwa wskazuje szyfruje tablicę bajtów, zgodnie z algorytmem AES Rijandel,

- Argumenty: tablica bajtów do zaszyfrowania, tablica bajtów klucza, tablica bajtów wektora inicjalizującego,
- Zwracane: zaszyfrowana tablica bajtów.

2.

```
public static byte [] DecryptBytes()
```

druga z kolei funkcja udostępniana przez bibliotekę AesLib, dokładnie antagoni-
styczna do pierwszej, czyli zajmująca się deszyfrowaniem danych,

- Argumenty: tablica bajtów do odszyfrowania, tablica bajtów klucza, tablica bajtów wektora inicjalizującego,
- Zwracane: odszyfrowana tablica bajtów.

3. Specyfikacja zewnętrzna

W tym rozdziale opisany zostanie sposób uruchomienia aplikacji oraz jej obsługa przez użytkownika. Zostaną także zawarte instrukcje postępowania w przypadku niepożądanego działania aplikacji

3.1 Sposób uruchomienia programu

Program pracuje w środowisku Windows. Aby poprawnie uruchomić aplikację należy posiadać wszystkie pliki otrzymane od wydawcy, a następnie należy dwukrotnie kliknąć w ikonkę "SpeakerRecognitionClient.exe". Program nie wymaga żadnej wcześniejszej konfiguracji. Jedynym wymogiem jest działający wbudowany mikrofon lub zewnętrzny wpięty do gniazda komputera oraz połączenie z internetem.

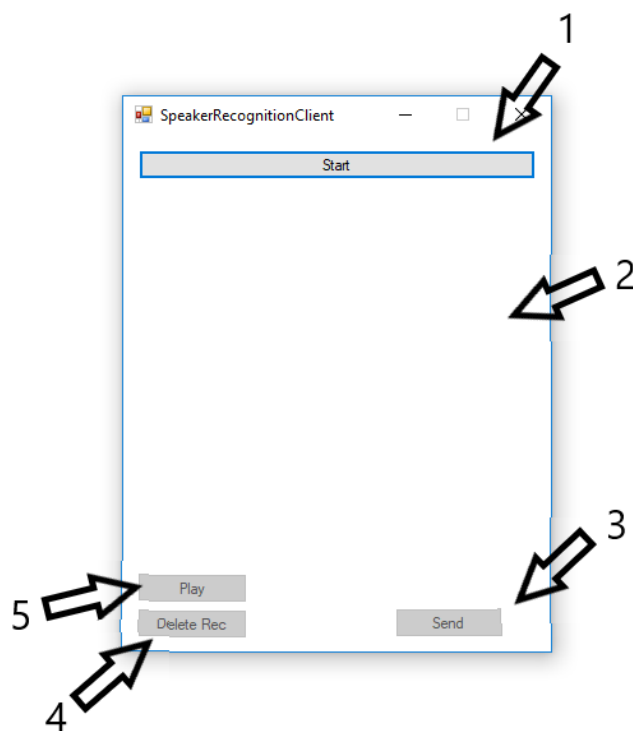
3.2 Obsługa programu

Po poprawnym uruchomieniu programu wyświetli się okno z prostym menu ekranu, które jest zarazem głównym i jedynym widokiem programu. Okno przedstawione jest na rysunku 3.1. Po wciśnięciu przycisku "Start" oznaczonego numerem 1 na rysunku 3.1 należy w przeciągu dwóch sekund wypowiedzieć słowo "BIOCYBERNETYKA" w celu nagrania próbki głosu. Istnieje możliwość odsłuchania nagranych głosu w celu sprawdzenia jakości nagrania. Odtworzenie głosu nastąpi po naciśnięciu przycisku "Play" oznaczonego numerem 5 na rysunku 3.1. Głos zostanie wysłany na serwer po wciśnięciu przycisku "Send" oznaczonego numerem 3 na rysunku 3.1. Istnieje także możliwość usunięcia nagrania poprzez wciśnięcie przycisku "Delete Rec" oznaczonego numerem 4.

Po wciśnięciu przycisku "Send" po bardzo krótkim czasie pojawi się decyzja dotycząca przydzielenia dostępu do systemu. Jeżeli dostęp został przydzielony, a więc osobą mówiącą jest osobą posiadającą uprawnienia do danego systemu, w polu oznaczonym numerem 2 na rysunku 3.1 pojawi się symbol informujący o pozytywnym rozpatrzeniu decyzji o dostępie. Okno po otrzymaniu dostępu przedstawione jest na rysunku 3.2.

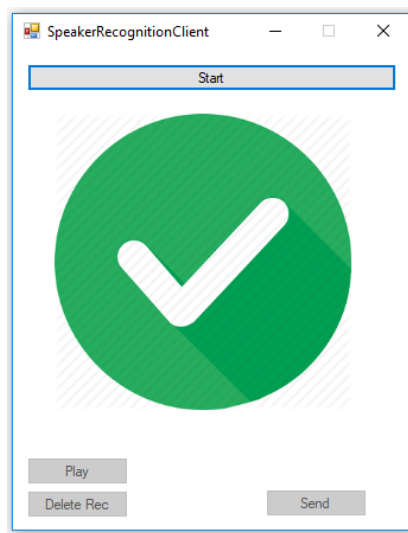
W przypadku braku dostępu do systemu, w polu oznaczonym numerem 2 na rysunku 3.1 pojawi się symbol informujący o braku dostępu. Okno po braku otrzymania dostępu przedstawione jest na rysunku 3.3.

Jeżeli nie otrzymano dostępu pomimo bycia osobą uprawnioną do niego, należy ponownie nagrać słowo testowe. Ważnym aspektem jest odpowiednia odległość ust od mikrofonu. W przypadku zbyt małej odległości zauważono nie poprawne działanie



Rys. 3.1: Główne okno aplikacji

algorytmu. Urządzenie nagrywające powinno znajdować się około 15 cm od ust osoby mówiącej.



Rys. 3.2: Okno informujące o dostępie do systemu

































Rys. 3.3: Okno informujące o braku dostępu do systemu

4. Testy

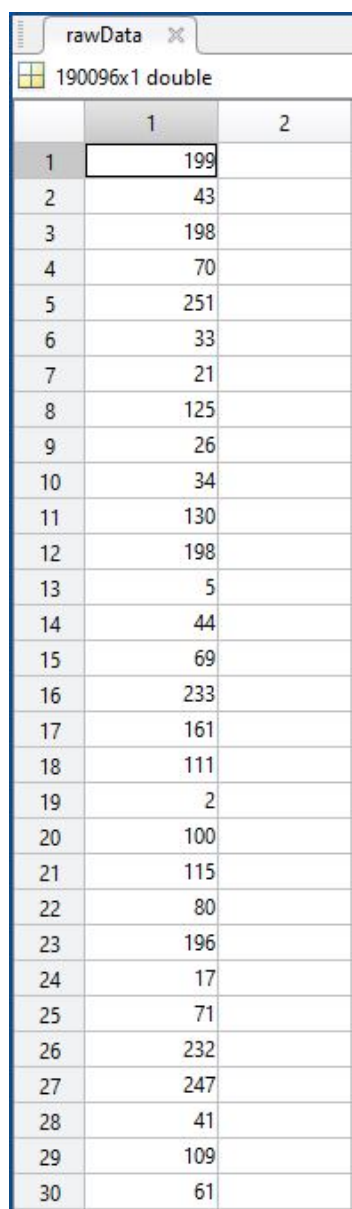
W tym rozdziale zostanie poruszony temat testowania możliwości oraz poprawności komunikacji klient-serwer. Protokół TCP/IP zapewnia integralność danych oraz, co było najważniejsze w wykonywanym projekcie, dostarczenie wszystkich wysyłanych danych. W każdym z testów dane wysyłane były identyczne do danych odbieranych na drugim końcu połączenia. Poprawność wysyłanych i odbieranych danych sprawdzono wykorzystując operację logiczną xor. Wykonanie tej operacji na identycznych zbiorach danych powoduje otrzymanie w wyniku samych zer. Taki też wynik otrzymano w testach. Dla wizualizacji na rysunku 4.1 oraz 4.2 przedstawiono 30 pierwszych bajtów danych po stronie serwera oraz klienta.

Jak widać dane są identyczne, co potwierdza że protokół TCP/IP zapewnia poprawną kolejność wymiany danych.

Kolejnym testem był test szybkości transmisji. Cechę tę testowana wykorzystując licznik czasu który uruchamiał się w momencie wysłania danych, a zatrzymywał się po odebraniu odpowiedzi od serwera. Czas danej transmisji można zobaczyć w prawym dolnym rogu okna przestawionego na rysunku 3.1. Faza testów wykazała, że wartość ta oscyluje wokół 250 ms.

 [0]	199
 [1]	43
 [2]	198
 [3]	70
 [4]	251
 [5]	33
 [6]	21
 [7]	125
 [8]	26
 [9]	34
 [10]	130
 [11]	198
 [12]	5
 [13]	44
 [14]	69
 [15]	233
 [16]	161
 [17]	111
 [18]	2
 [19]	100
 [20]	115
 [21]	80
 [22]	196
 [23]	17
 [24]	71
 [25]	232
 [26]	247
 [27]	41
 [28]	109
 [29]	61

Rys. 4.1: 30 pierwszych bajtów wysyłanych po stronie klienta



	1	2
1	199	
2	43	
3	198	
4	70	
5	251	
6	33	
7	21	
8	125	
9	26	
10	34	
11	130	
12	198	
13	5	
14	44	
15	69	
16	233	
17	161	
18	111	
19	2	
20	100	
21	115	
22	80	
23	196	
24	17	
25	71	
26	232	
27	247	
28	41	
29	109	
30	61	

Rys. 4.2: 30 pierwszych bajtów odebranych po stronie serwera

5. Podsumowanie

Jak pokazały testy udało się stworzyć aplikację która w szybki, a co najważniejsze, bez naruszenia integralności danych sposób, transmituje ludzką mowę. Zrealizowano w ten sposób cel projektu, choć nie zachowały się pierwotne założenie. Zakładano wtedy, że do połączenia klienta z serwerem skorzysta się z technologii *Windows Communication Foundation*. Zrezygnowano z tego pomysłu, ponieważ wczesne testy wykazały, że przesył danych trwał zbyt długo.

Główną wadą aplikacji jest niestałość w przysyłaniu wszystkich danych, lecz błąd ten, jak wykazały testy, dotyczy poszczególnych komputerów i nie jest bezpośrednio związany z wadliwie działającą aplikacją. Błąd ten może być powiązany z antywirusem z którego korzysta użytkownik. Jest to kwestia której należy przyjrzeć, aby usprawnić działanie aplikacji.

Kolejnym usprawnieniem aplikacji może być przyspieszenie przesyłu danych. Aktualnie aplikacja wysyła surowe dane dźwiękowe które następnie są przetwarzane przez serwer. Takie rozwiązanie wymaga przesłania bardzo dużej ilości danych które są następnie znacznie zredukowane na poziomie serwera. Istnieje możliwość wykonywania części obliczeń przez urządzenie użytkownika (pozyskiwanie współczynników MFC) co znacznie zredukuje ilość danych przesyłanych na serwer, a tym samym przyspieszy działanie aplikacji.

Ważnym aspektem w kierunku którego powinna rozwinać się aplikacji jest możliwość połączenie się wielu klientów jednocześnie.