

POLITECHNIKA ŚLĄSKA
WYDZIAŁ INŻYNIERII MATERIAŁOWEJ I METALURGII

Kierunek: Informatyka Przemysłowa
Rodzaj studiów: Stacjonarne I stopnia

PROJEKT INŻYNIERSKI

Tomasz Żmijowski

Temat projektu inżynierskiego

**PROJEKT I WDROŻENIE APLIKACJI
POWIADOMIEŃ SMS DO ZASTOSOWANIA
W SYSTEMIE INFORMATYCZNYM
W PRZEDSIĘBIORSTWIE I BIZNESIE**

**DESIGN AND IMPLEMENTATION OF AN SMS
NOTIFICATION SYSTEM FOR INDUSTRIAL AND
BUSINESS IT SOLUTIONS**

Kierujący projektem:
Dr inż. Marcin Blachnik

Recenzent:
Dr inż. Paweł Światała

Opiekun merytoryczny projektu:
Dr Marcin Pilarczyk

Katowice, styczeń 2014 r.

Spis treści

1. Wstęp	5
1.1 Cele pracy	5
1.2 Struktura pracy	6
2. Standard SMS	7
2.1 Format i wielkość wiadomości	7
2.2.1 Format PDU	8
3. Komunikacja z urządzeniem wykonawczym	9
3.1 Standard Hayes'a	9
3.2 Zbiór poleceń AT	9
3.2.1 Komendy podstawowe	9
3.2.2 Komendy rozszerzone	10
3.2.3 Składnia poleceń	10
3.2.4 Odpowiedź i kod błędu	11
3.3 Sposób sterowania komendami AT	12
3.4 Wysyłanie wiadomości SMS poprzez aplikację typu Terminal	12
3.4.1 Wykorzystanie trybu SMS TEXT	12
3.4.2 Wykorzystanie trybu PDU	13
4. System powiadomień SMS	14
4.1 Zastosowanie systemu powiadomień w przedsiębiorstwie i biznesie	14
5 Projekt systemu powiadomień SMS	15
5.1 Serwer SMS - SMSS	15
5.1.1 Zbieranie wymagań	15
5.1.2 Dobór technologii	16
5.1.3 Wymagania systemowe i sprzętowe	16
5.1.4 Wykorzystanie zewnętrznych bibliotek programistycznych	17
5.1.5 Zaimplementowane funkcjonalności	17
5.1.6 Podgląd działania aplikacji	22
5.2 Aplikacja kliencka - SMSC	23
5.2.1 Zbieranie wymagań	23

5.2.2 Dobór technologii	23
5.2.3 Wymagania systemowe i sprzętowe	24
5.2.4 Wykorzystanie zewnętrznych bibliotek programistycznych	24
5.2.5 Zaimplementowane funkcjonalności	25
5.2.6 Wykorzystanie bazy danych	30
5.2.7 Bezpośrednie odwoływanie się do SMSC poprzez Socket.....	32
5.2.8 Podgląd działania aplikacji	33
6 Podsumowanie i wnioski	34
Spis literatury	35
Spis rysunków, tabel i kodów	36
Tabele.....	36
Rysunki	36
Kody.....	36

1. Wstęp

Znaczenie informacji w dzisiejszych czasach ma bardzo duże znaczenie. Informacje znajdują się wszędzie, w gazetach, telewizji czy Internecie. Jednakże nie każdy komunikat dla odbiorcy jest wartościowy, a co gorsza, ich ilość często sprawia że można przeoczyć istotną informację. W przedsiębiorstwach jak i biznesie, oprócz samej treści wiadomości, znaczenie ma również czas w jakim wiadomość do nas dotrze. Czas ma często bardzo krytyczne znaczenie, np. informacja o awarii systemu komputerowego w firmie, odebrana przez technika kilka godzin po zaistniałej sytuacji, może przynieść znaczące straty dla przedsiębiorstwa. Gdyby informacja o potencjalnym zagrożeniu została przekazana do odpowiedniej osoby wcześniej, to być może nie doszłoby do strat. Pytanie brzmi jak system komputerowy może nas poinformować? Bardzo popularny dzisiaj kanał komunikacji jakim jest SMS (Short Message Service), w połączeniu z systemem powiadomień sprawia, że można przesłać spersonalizowaną informację prosto do naszego telefonu w bardzo krótkim czasie. W ten sposób zarówno pracownicy odpowiedzialni za utrzymanie systemów mogli by być poinformowani o awarii, jak i klienci korzystający z usług otrzymaliby informacje o chwilowych problemach technicznych.

1.1 Cele pracy

Celem pracy było stworzenie systemu powiadomień, mającego zdolność wysyłania krótkich wiadomości tekstowych (SMS) – marketingowych, informacyjnych, różnego rodzaju powiadomień, kodów jednorazowych i innych. Odbiorcami tych wiadomości z założenia byłoby klienci jak i pracownicy przedsiębiorstwa.

Motywacją do obrania tego tematu pracy, była chęć stworzenia systemu powiadomień na potrzeby lokalnej sieci komputerowej w której przyszło mi administrować.

Należy zaznaczyć, że ze względu na różnorodność systemów informatycznych stosowanych w przedsiębiorstwach, nie jest się w stanie stworzyć uniwersalnego systemu powiadomień. Wdrożenie takiego systemu, wymaga indywidualnego podejścia i dopasowania pod kątem funkcjonowania.

1.2 Struktura pracy

Rozdział 2 zawiera podstawy teoretyczne niezbędne do zrozumienia działania i realizacji usług SMS.

W rozdziale 3 zaprezentowane zostały przykładowe komendy AT wykorzystywane do realizacji komunikacji z urządzeniem wykonawczym, którym może być centralka telefoniczna, modem, aparat telefoniczny, telefon komórkowy itp. Standard ten wprowadzony przez firmę Hayes, jest ogólnie przyjętym rozwiązaniem i implementowany jest w większości urządzeń telekomunikacyjnych.

W rozdziale 4 zostały przedstawione podstawowe informacje i cechy systemów pełniących funkcje serwerów SMS. Przyjmuje się założenia w oparciu o które będzie on funkcjonował, oraz korzyści jakie będzie przynosił.

Rozdział 5 zawiera założenia i przedstawia kolejne etapy realizację projektu serwera SMS (SMSS) w oparciu o język programowania JAVA, oraz klienta SMSC będącego interfejsem użytkownika, stworzonego w języku PHP.

Realizowany projekt składa się z trzech wzajemnie uzupełniających się elementów:

- aplikacji której zadaniem jest wysyłanie wiadomości SMS poprzez urządzenie podłączone do komputera na którym ta aplikacja jest uruchomiona,
- aplikacji będącej interfejsem użytkownika,
- bazy danych.

2. Standard SMS

SMS (Short Message Service) jest usługą oferowaną w cyfrowych sieciach telefonii komórkowej jak i sieciach stacjonarnych, której celem jest przesyłanie krótkich wiadomości tekstowych. SMS jest jednym z najczęściej wykorzystywanych kanałów informacyjnym, zaraz po wiadomościach przesyłanych poprzez komunikatory internetowe i emaile.

Historia krótkich wiadomości tekstowych sięga lat 80, kiedy to padła koncepcja opracowania standardu. W 1992 roku został wysłany pierwszy SMS przez pracownika firmy Vodafone w którym to życzył współpracownikom Wesołych Świąt. Początkowo krótkie wiadomości tekstowe były wykorzystywane przez operatorów sieci GSM do diagnostyki sieci, przesyłaniu komunikatów technicznych czy informowaniu klientów o zmianach w sieci.

2.1 Format i wielkość wiadomości

Długość wiadomości SMS została wyznaczona statystycznie na podstawie ilości słów pisanych na popularnych kartach pocztowych i wyniosła 150 znaków. Do tego wyniku dodane zostało zapasowe 10 znaków i w wyniku tego standardowa długość wiadomości SMS została ustalona na 160 znaków przy kodowaniu 7-bitowym. Standardowe kodowanie GSM nie zawiera polskich znaków diakrytycznych, - ą, ć, ę, ł, ń, ó, ś, ź, ż. Dlatego też w przypadku ich stosowania, wymusza się kodowanie 16-bitowe, w wyniku czego długość wiadomości spada do 70 znaków. [5]

Ilość bitów na znak	Maksymalna długość wiadomości	Zastosowanie
7	160 znaków	Wiadomości zawierające standardowe znaki z tablicy kodów ASCII
8	140 znaków	Stosowane w przypadku wysyłania wiadomości typu vCard i konfiguracji telefonu przez operatorów
16	70 znaków	Wiadomość zawierająca niestandardowe znaki (ą, ć, ę itp) kodowania UNICODE UTF-16

Tabela 1 Zestawienie ilości znaków w wiadomościach SMS

W przypadku konieczności przesłania dłuższych wiadomości stosuje się usługę SMSC (Concentrated SMS), która to automatycznie dzieli oryginalną wiadomość na części, a po wysłaniu składa w całość po stronie odbiorcy.

2.2.1 Format PDU

PDU (Protocol Description Unit) jest formatem pozwalającym w optymalny sposób wykorzystać dostępne dla wiadomości miejsce, jednak wiąże się to ze skomplikowaniem procesu tworzenia treści wiadomości. Kodowanie PDU opiera się o zapis kodów znaków ASCII na 7 bitach. Wiadomość SMS, oprócz treści zawiera zbiór informacji (tzw. meta-informacje), niezbędnych do wysłania wiadomości. Należą do nich między innymi: numer centrum wiadomości (SMSC), numer i nazwa nadawcy, data i godzina nadania, rodzaj kodowania. [5,8]

Przykładowa wiadomość:

- treść wiadomości: Co u Ciebie?,
- numer odbiorcy: 31641600986,
- numer SMSC: 31624000000

Wiadomości w formacie PDU:

07911326040000F011000B911346610089F60000AA0CC337A80E1AA6CBE274F907
--

Znaczenie poszczególnych oktetów:

Oktet	Opis	Wartość
07	Długość informacji SMSC	7
91	Typ adresu SMSC	...
13 26 04 00 00 F0	Numer SMSC	31624000000
04	Pierwszy oktet SMS-DELIVERY PDU	TP-MMS
0B	Długość adresu nadawcy	11
91	Typ adresu numeru nadawcy	...
13 46 61 00 89 F6	Numer odbiorcy	31641600986
00	Identyfikator protokołu {PID}	00
00	Schemat kodowania danych (DCS)	00
20 80 62 91 73 14 08	Znacznik czasu	06-08-02 29:17:31
0C	Długość wiadomości	12
C3 37 A8 0E 1A A6 CB E2 74 F9 07	Wiadomość	Co u Ciebie?

Tabela 2 Znaczenie poszczególnych oktetów wiadomości zapisanej w formacie PDU

3. Komunikacja z urządzeniem wykonawczym

3.1 *Standard Hayes'a*

Hayes jest standardem komunikacji opracowanym przez firmę Hayes Microcomputer Products, który to wkrótce stał się ogólnie-przyjętym sposobem komunikacji z modemami i urządzeniami telefonicznymi. Pierwotnie służył on komunikacji z modemami analogowymi, lecz został on również zaimplementowany do urządzeń GSM – telefonów i modemów komórkowych.

AT jest skrótem słowa „attention” czyli uwaga. Prefiks ten dodawany przed poleceniem informuje urządzenie o nadchodzącym rozkazie. Mimo tego, że w standardzie zostały jasno zdefiniowane poszczególne polecenia, producenci sprzętu często wprowadzają modyfikacje, bądź implementują własne zbiory instrukcji, a ich możliwość realizacji zależy od funkcji dostępnych w urządzeniu lub usług oferowanych przez operatora sieci komórkowej. Ze względu na to że każde polecenie zaczyna się do liter „AT”, standard często nosi nazwę komend AT. Producenci sprzętu (najczęściej telefonów komórkowych) często nie implementują wszystkich komend AT, ograniczając możliwość sterowania i zmian zaawansowanych parametrów urządzenia. [8, 10, 11]

3.2 *Zbiór poleceń AT*

3.2.1 Komendy podstawowe

Istnieją dwa rodzaje komend. Podstawowe nie zawierają operatora „+” pomiędzy prefiksem AT a poleceniem. [6,7,11]

Przykłady podstawowych komend AT:

Polecenie	Opis
ATD	Wybranie numeru telefonu
ATA	Odebranie połączenia
ATH	Rozłączenie połączenia
ATO	Powrót do trybu przesyłania danych
+++	Przełączenie w tryb rozkazowy
A/	Powtórzenie ostatniego polecenia
AT	Zwraca OK – komenda kontrolna

Tabela 3 Zestawienie przykładowych komend AT (Basic AT Command Set)

3.2.2 Komendy rozszerzone

Komendy rozszerzone zawierają po prefiksie operator „+”. Każda komenda Hayes’a rozpoczyna się prefiksem AT po którym następuje polecenie. [6,7,11]

Przykład popularnych komend AT:

Polecenie	Opis
AT+CGMI	Zwraca producenta aparatu
AT+CGMM	Zwraca nazwę aparatu
AT+CBC	Zwraca informacje o stanie naładowania baterii
AT+CIMI	Zwraca numer IMSI
AT+CGSN	Zwraca numer IMEI
AT+CMGF=0/1	Ustawienie trybu tekstowego/PDU wiadomości
AT+CMGL	Zwraca listę wiadomości SMS

Tabela 4 Zestawienie przykładowych komend AT (Extended AT Command Set)

3.2.3 Składnia poleceń

Na składnię komend składa się kilka reguł których nieprzestrzeganie skutkuje komunikatem błędu [8]:

- Każde polecenie musi zaczynać się od prefiksu „AT” i kończyć się znakiem „powrotu karetki” <CR>.

AT<CR> OK
AT+CMGF=1<CR> OK

- b) Polecenia można wydawać w jednym wierszu przedzielając je średnikami, lecz prefiks podaje się tylko jednorazowo na początku polecenia.

```
AT+CGMI;+CGMM<CR>
```

- c) Ciągi znakowe jako wartości polecenia należy umieszczać w cudzysłowach.

```
AT+CMGL="ALL"<CR>
```

- d) Nie należy stosować naprzemiennie dużych i małych liter w poleceniu, ponieważ w zależności od urządzenia może on zwrócić błąd.

```
AT+CMGF=1<CR>           //przykład poprawnej składni  
at+cmgf=1<CR>           //przykład poprawnej składni
```

```
At+Cmgf=1<CR>           //przykład błędnej składni
```

3.2.4 Odpowiedź i kod błędu

Każde polecenie zadane urządzeniu skutkuje informacją zwrotną. W przypadku pomyślnego wykonania polecenia urządzenie zwraca OK, w przypadku błędu ERROR [8].

```
AT+CGMI //polecenie - identyfikacja producenta urządzenia  
Nokia //odpowiedź zwrotna  
  
OK //kod błędu  
  
AT+CGMM //błędne polecenie  
ERROR //kod błędu
```

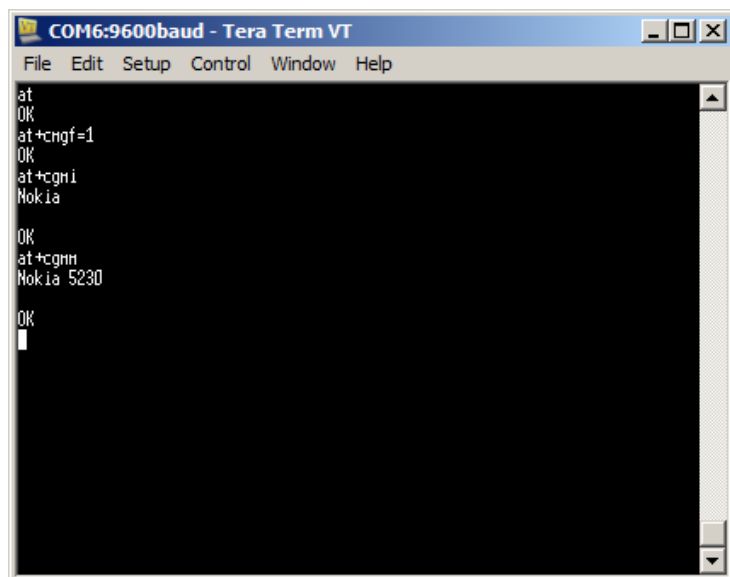
Kod błędu może pojawić się w sytuacji gdy:

- składnia polecenia była niepoprawna
- wartość parametru była nieprawidłowa,
- polecenie nie było obsługiwane przez urządzenie.

3.3 Sposób sterowania komendami AT

Polecenia do urządzenia przekazuje się poprzez aplikację (terminal).

Przykłady programów do realizacji transmisji szeregowej dla systemu Windows: Hyper Terminal, Putty, Tera Term. Przykłady programów do realizacji transmisji szeregowej dla systemu Linux/Unix: minicom, cu, putty, screen. [7]



Rysunek 1 Przykład wykorzystania komend AT programem Tera Term

Transmisję szeregową można również zrealizować tworząc własne oprogramowanie, wykorzystując biblioteki programistyczne dostępnych na wszelkie znane i używane języki programowania: C, C++, C#, Java.

3.4 Wysyłanie wiadomości SMS poprzez aplikację typu Terminal

Standard AT definiuje dwa rodzaje wysyłania wiadomości. Tryb TEXT i PDU. Główna różnica pomiędzy nimi wynika z faktu że tryb PDU umożliwia definiowanie nagłówku wiadomości, gdzie w przypadku trybu TEXT, wykorzystywane są wartości domyśle znajdujące się w konfiguracji modemu.

3.4.1 Wykorzystanie trybu SMS TEXT

W trybie tym wymagane jest jedynie podanie numeru odbiorcy i treści wiadomości. Wszelkie dodatkowe informacje wymagane do utworzenia nagłówka wiadomości zostaną pobrane z konfiguracji modemu.

Składnia:

```
AT+CMGW=numer_odbiorcy<CR>treść_wiadomości<Ctrl+z>
```

Przykład:

Numer odbiorcy: +48555666222

Treść wiadomości: Moja wiadomość

```
AT+CMGF=1<CR> //ustawienie trybu tekstowego
AT+CMGS="+48555666222"<CR> //polecenie wysłania wiadomości
Moja wiadomość.<CTRL+Z> //treść wiadomości i zatwierdzenie poprzez
CTRL+Z
```

3.4.2 Wykorzystanie trybu PDU

trybie PDU do wysłania wiadomości wymagane jest podanie informacji nt. centrum SMS (SMSC) i wygenerowanie ciągu zapisanego w formacie Heksadecymalnym (HEX). [8]

Składnia:

```
AT+CMGW=TPDU_długość<CR>SMSC_i_treść_wiadomość<Ctrl+z>
```

Przykład:

Numer odbiorcy: +48555666222

Treść wiadomości: Moja wiadomość

Centrum SMS: +48790998250

```
AT+CMGF=0<CR> //ustawienie trybu PDU
AT+CMGS=27<CR> //polecenie wysłania wiadomości
07918497908952F01100099144556677F80004AA0E4D6F6A6120776961646F6D6F5B07
<CTRL+Z> //treść wiadomości i zatwierdzenie poprzez CTRL+Z
```

4. System powiadomień SMS

System powiadomień jest specjalistycznym oprogramowaniem, odpowiedzialnym za wysyłanie wiadomości SMS poprzez urządzenie podłączone do tego systemu – może nim być telefon komórkowy, modem GSM itp., Wysyłanie wiadomości SMS następuje w sposób automatyczny po zdefiniowaniu zadania poprzez aplikację kliencką. W skład systemu powiadomień wchodzi:

- aplikacja realizująca komunikację z urządzeniem wykonawczym - serwer SMS,
- interfejs użytkownika – klient SMS (web-aplikacja, aplikacja desktopowa, skrypt, interfejs-api),
- urządzenie wykonawcze (telefon komórkowy, modem GSM itp.).

4.1 Zastosowanie systemu powiadomień w przedsiębiorstwie i biznesie

Ze względu na motyw implementacji systemu można wyróżnić dwie podstawowe grupy odbiorców systemu powiadomień:

- Klienci i odbiorcy usług:
 - definiowanie powiadomień dla klientów – wszelkiego rodzaju powiadomienia dla klientów końcowych – firmy handlowe, transport i spedycja, informacje o dostępności lub niedostępności towaru/usług,
 - masowa wysyłka SMS – wysyłanie masowej ilości wiadomości do zdefiniowanych odbiorców,
 - przypomnienia o spotkaniach,
 - usługa Email to SMS,
 - konkursy i ankiety SMS.
- Pracownicy firmy:
 - monitorowanie pracy urządzeń,
 - sterowanie pracą urządzeń,
 - autoryzacja i uwierzytelnianie użytkowników za pomocą haseł jednorazowych.

5 Projekt systemu powiadomień SMS

W skład systemu powiadomień wchodzi dwie powiązane ze sobą aplikacje:

- Serwer SMS – SMSS – aplikacja odpowiedzialna głównie za komunikację z urządzeniem wykonawczym i wysyłanie wiadomości. Serwer został napisany w języku programowania JAVA.
- Klient SMS – SMSC – aplikacja będąca interfejsem użytkownika. Web-aplikacja zapewniająca wygodny sposób definiowania i planowania zadań dla SMSS.

Nieodłącznym elementem system jest również baza danych, która łączy obie aplikacje, stanowi magazyn zadań dla serwera, przechowuje historię zadań oraz informacje o użytkownikach i kontaktach. Oprócz standardowego gromadzenia nowych zadań w bazie danych, istnieje możliwość zlecenia zadań wykorzystując bezpośrednie odwoływanie się do aplikacji serwerowej w architekturze klient-serwer poprzez połączenie typu Socket. Rozwiązanie takie ma też dodatkowy plus – aplikacja serwerowa działa jako usługa (w tle), w wyniku czego nie ma możliwości sterowania – połączenie typu Socket sprawia że można sterować SMSS z poziomu innej aplikacji.

5.1 Serwer SMS - SMSS

5.1.1 Zbieranie wymagań

Wymagania funkcjonalne

- Komunikacja z urządzeniem wykonawczym,
- Wysyłanie wiadomości poprzez urządzenie wykonawcze,
- Obsługiwanie wyjątkowych sytuacji i próby ponownego wysłania wiadomości w przypadku problemów z urządzeniem/siecią,
- Tworzenie historii działań,
- Udostępnienie kanału komunikacji z zewnętrznym oprogramowaniem.

Wymagania pozafunkcjonalne

- Niezawodność
 - Aplikacja nie powinna zaburzać pracy systemu operacyjnego,

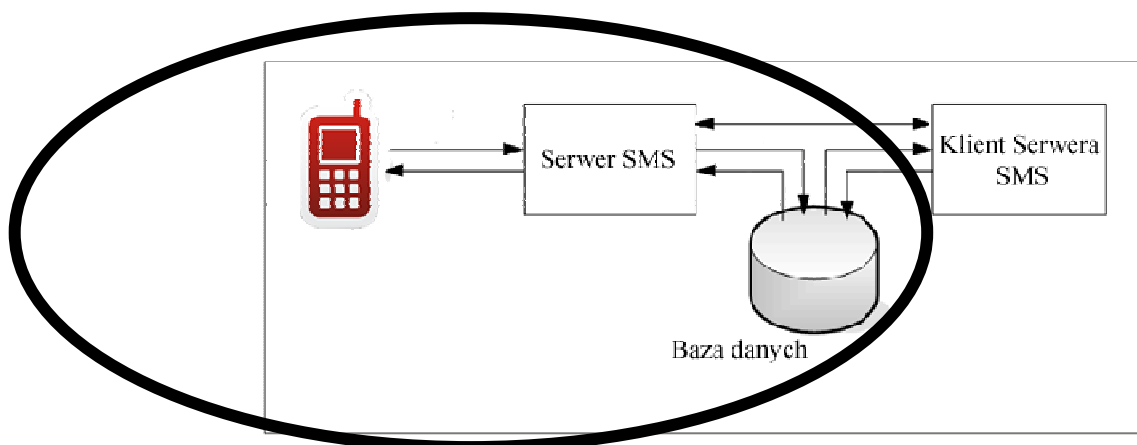
- W przypadku wystąpienia błędu powinna go obsługiwać i nie powodować zakończenia działania aplikacjami,
- Aplikacja powinna poprawnie współpracować z zewnętrznym oprogramowaniem.
- Wydajność
 - Zdolność aplikacji do szybkiego wysyłania wiadomości bez zbędnych przerw.
- Wielowątkowość
 - Zdolność aplikacji do wykonywania kilku zadań jednocześnie – m.in. wysyłania wiadomości, komunikacja z bazą danych, obsługiwanie żądań innych aplikacji poprzez Socket.
- Wieloplatformowość
 - Aplikacja powinna być wieloplatformowa – zdolność uruchomienia na różnych systemach operacyjnych.

5.1.2 Dobór technologii

Na wybór języka programowania i sposobu realizacji miały wpływ wymagania projektu, między innymi wieloplatformowość – możliwość uruchomienia na wielu różnych systemach operacyjnych oraz własne doświadczenie w pisaniu aplikacji w języku JAVA.

5.1.3 Wymagania systemowe i sprzętowe

W celu uruchomienia aplikacji SMSS wymagane jest posiadanie środowiska uruchomieniowego JAVA – JRE (JAVA Runtime Environment), w wersji nie niższej niż 7. Konieczne będzie również posiadania urządzenia wykonawczego z zaimplementowaną możliwością komunikacji z wykorzystaniem komend AT oraz bazy danych. W projekcie została wykorzystana baza MySQL w wersji 5.0.10 (część pakietu XAMPP opisanego w wymaganiach aplikacji SMSC).



Rysunek 2 Zakres komunikacji aplikacji Serwer SMS

5.1.4 Wykorzystanie zewnętrznych bibliotek programistycznych

Realizacja transmisji szeregowej została użyta za pomocą biblioteki RxTx autorstwa Trent Jarvi, rozpowszechnianej na licencji LGPL.

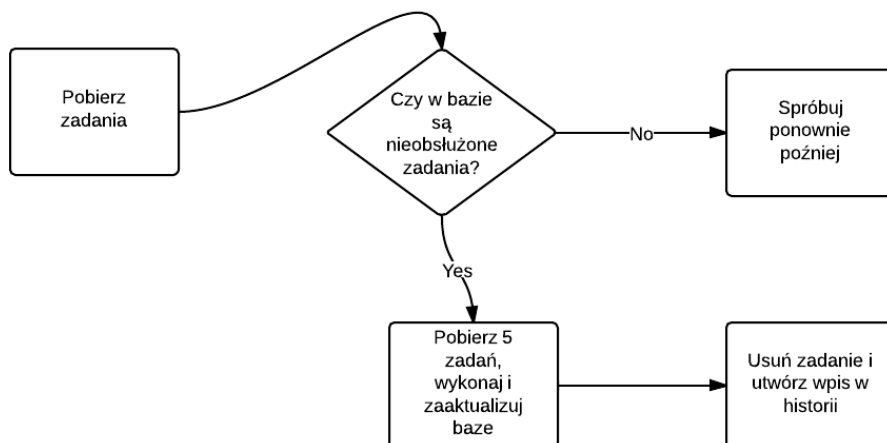
Kolejny zbiór klas to biblioteka google-gson wersja 2.2.4, odpowiedzialna za konwertowanie obiektów Java do ciągów znakowych JSON i odwrotnie. Ma ona zastosowanie przy realizacji połączeń poprzez Socket – wszelkie dane przychodzące powinny być przesyłane w tego typu ciągach.

Ostatnią dołączoną biblioteką jest sterownik bazy danych MySQL w wersji 5.1.27.

5.1.5 Zaimplementowane funkcjonalności

5.1.5.1 Rozwiązanie buforowania wiadomości

Jako główne źródło zadań do realizacji stanowi baza danych, do której aplikacje klienckie zapisują nowe zadania. Aplikacja SMSS w trakcie działania odpytuje bazę danych i pobiera do własnego bufora ustaloną ilość nowych zadań. Domyślny rozmiar bufora wewnętrznego aplikacji wynosi pięć. Tego typu rozwiązanie ma przyczynić się do zminimalizowania ilości zapytań do bazy. W przypadku kiedy w bazie pojawiają się nowe zadania, aplikacja SMSS oznacza określoną ilość zadań, nadając im status będącej w realizacji. Bufor działa w sposób – pierwszy wchodzi, pierwszy wychodzi (FIFO). Po wykonaniu zadania przez SMSS, wpis w bazie jest usuwany, a zaimplementowany wyzwalacz w bazie danych tworzy nowy rekord w tabeli zawierającej historie.



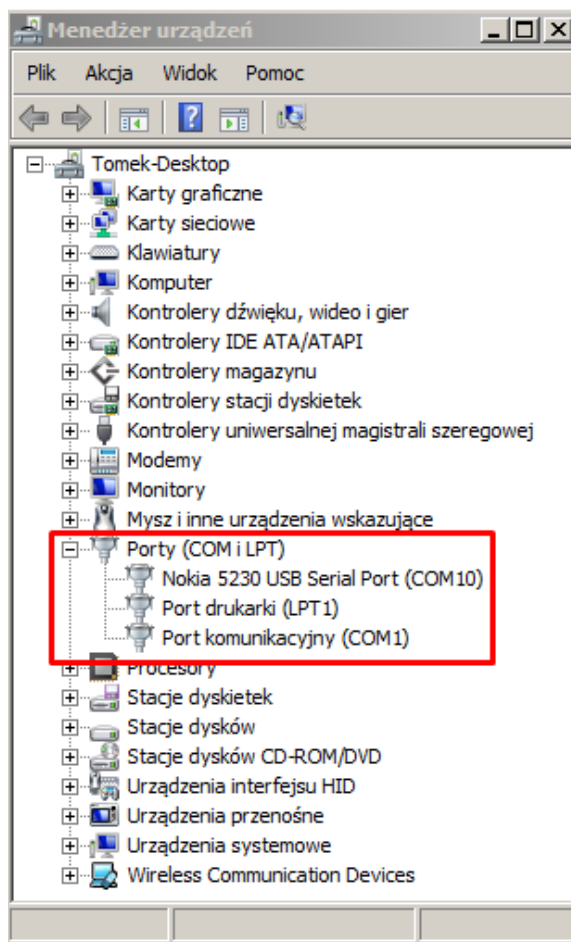
Rysunek 3 Schemat obsługi zadań

5.1.5.2 Interfejs komunikacji z zewnętrznymi aplikacjami

W aplikacji SMSS zaimplementowana została możliwość odwołania się do programu poprzez połączenie bezpośrednie – Socket z wykorzystaniem portu i adresu IP na jakim działa aplikacja. Z uwagi że jest to aplikacja konsolowa, z myślą funkcjonowania w środowisku tekstowym, w tle, nie ma zbyt szerokiego pola manewru jeżeli chodzi o interakcje z aplikacją. Zaimplementowanie możliwości komunikacji powoduje że można sterować pracą aplikacji.

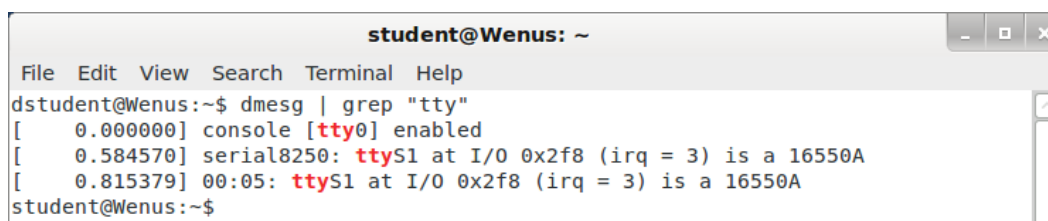
5.1.5.3 Wyszukiwanie i komunikacja z urządzeniem

W zależności od systemu operacyjnego urządzenia wykonawcze są reprezentowane w systemie operacyjnym w różny sposób. W systemach z rodziny Windows są one przedstawiane jako urządzenia **COMx**, gdzie **x** to numer portu. Informacje na temat podłączonych urządzeń można znaleźć w Menadżer urządzeń.



Rysunek 4 Podgląd dostępnych portów COM - Windows

W systemach z rodziny Linux porty szeregowo widziane są jako `/dev/ttySx`, gdzie `x` to numer portu. Dostępne porty można sprawdzić poleceniem `dmesg | grep "tty"`.



Rysunek 5 Podgląd dostępnych portów COM - Linux

Wykorzystana biblioteka RxTx jest w stanie funkcjonować zarówno w systemach Windows jak i Linux z powodu czego, dostęp do urządzeń został w znaczny sposób uproszczony. [1,9]

```
//wykrywanie portow COM w systemie, zapis do listy
public void searchForPorts() {
    ports = null;
    ports = CommPortIdentifier.getPortIdentifiers();

    while (ports.hasMoreElements()) {
        CommPortIdentifier curPort = (CommPortIdentifier)ports.nextElement();

        if (curPort.getPortType() == CommPortIdentifier.PORT_SERIAL) {
            ListaPortow.add(curPort);
        }
    }
}
```

Listing kodu 1 Wykrywanie dostępnych portów COM

5.1.5.4 Wykorzystanie bazy danych jako zbioru zadań

Zastosowanie bazy danych jako centralnego punktu przechowywania zadań niesie za sobą następujące korzyści:

- W razie uruchomienia kilku instancji aplikacji do obsługi większej ilości urządzeń, można dynamicznie rozdzielać zadania poprzez rezerwacje ich w bazie danych,
- Możliwość dodawania zadań poprzez skrypty systemowe czy inne aplikacje poprzez dodanie rekordu do bazy,
- W przypadku gdy aplikacja będzie wyłączona lub działanie zostanie zawieszone, do czasu jej funkcjonowania zadania będą dodawane do bazy.

5.1.5.5 Proces wysyłania wiadomości

W momencie gdy SMSC wyśle żądanie, a aplikacja umieści je w bazie, rozpoczyna się proces wysyłania wiadomości. W pierwszej kolejności SMSS odpytuje bazę o nowe zadania. Jeżeli się pojawią to markuje je i pobiera do własnego bufora. Zadania w bazie przechowywane są w obiektach typu JSON. Po przetworzeniu ciągu znakowego, następuje tworzenie wiadomości SMS. Odpowiada za to metoda tworząca listę typu *string* generując gotową komendę przeznaczoną dla urządzenia wykonawczego. [1,8,11]

```

protected ArrayList<String> PrepareMessage(String number, String message) {
    //znak 23 End of Transmission Block ETB ctrl+z
    ArrayList<String> command = new ArrayList<String>();
    command.add(""+(char)27); //escape, czyszczenie terminala
    command.add("AT\r");
    command.add("AT+CMGF=1\r");
    command.add("AT+CMGS=\""+number+"\"\\r");
    command.add(message+(char)26);
    return command;
}

```

Listing kodu 2 Tworzenie komendy wysyłającej wiadomość SMS

Po przygotowaniu polecenia następuje wywołanie metody zapisu danych do urządzenia i jednocześnie uruchamiany jest nasłuch w celu odebrania informacji zwrotnej.

5.1.5.6 Znaczenie wielowątkowości

Wymagane jest aby aplikacja SMSS realizowała w jednym czasie wiele operacji – komunikowała się z bazą danych, nasłuchiwała połączeń TCP, i wysyłała na bieżąco wiadomości. Dzięki wątkom można zapewnić wykonywanie się wielu zadań równocześnie.

5.1.6 Podgląd działania aplikacji

```
Input - smsApp (run) X
run:
URUCHAMIANIE PROGRAMU...
#####
ŁADOWANIE KONFIGURACJI...
Plik konfiguracyjny istnieje... C:\Users\Tomek\Documents\NetBeansProje
    DB HOST: 127.0.0.1
    DB USER: sms
    DB PASS: sms
    DB NAME: sms
    Socket Port: 9990
    Device COM Port: COM6
#####
TESTOWANIE KONFIGURACJI
-> Wyszukiwanie portów COM w systemie...
Stable Library
=====
Native lib Version = RXTX-2.2-20081207 Cloudhopper Build rxtx.cloudhop
Java lib Version   = RXTX-2.1-7
WARNING: RXTX Version mismatch
    Jar version = RXTX-2.1-7
    native lib Version = RXTX-2.2-20081207 Cloudhopper Build rxtx.
#####
-> Testowanie komunikacji z urządzeniem...
    out: AT
    out: OK
    Nazwa urządzenia:
    out: AT+GMM
    out: Nokia 5230
    out: OK
-> Uruchamianie wątku przydzielania zadań...
-> Reset rezerwacji zadań w bazie danych...
Main END
-> Uruchamianie wątku nasłuchowania połączeń przychodzących...
    Uruchamianie socketu na porcie:9990
-> Oczekiwanie na klienta...
-> Brak zadań w bazie danych... sleep(5s)
-> Brak zadań w bazie danych... sleep(5s)
-> Brak zadań w bazie danych... sleep(5s)
```

Rysunek 6 Log uruchamiania się aplikacji

```

-> Brak zadan w bazie danych... sleep(5s)
-> Brak zadan w bazie danych... sleep(5s)
    -> Zad do bufora:{"id_task":"6","message":"Piękny mamy dzisiaj dzień!","number":"██████████"}
        out: AT
        out: OK
        out:
        out: AT
        out: OK
        out: AT+CMGF=1
        out: OK
        out: AT+CMGS="692215875"
        out:
        out: Piękny mamy dzisiaj dzień!
-> Brak zadan w bazie danych... sleep(5s)

```

Rysunek 7 Log wysyłania wiadomości

```

-> Brak zadan w bazie danych... sleep(5s)
=> Polaczenie przychodzace... Fri Jan 10 00:57:18 CET 2014/127.0.0.1
-> Oczekiwanie na klienta...
-> Brak zadan w bazie danych... sleep(5s)

```

Rysunek 8 Log komunikacji poprzez Socket

5.2 Aplikacja kliencka - SMSC

5.2.1 Zbieranie wymagań

Wymagania funkcjonalne

- definiowanie zadań dla serwera SMS,
- zarządzanie użytkownikami,
- tworzenie bazy klientów.

Wymagania pozafunkcjonalne

- użyteczność,
- niezawodność,
- ogólnodostępność,
- wieloplatformowość,
- łatwa możliwość dalszego rozwoju.

5.2.2 Dobór technologii

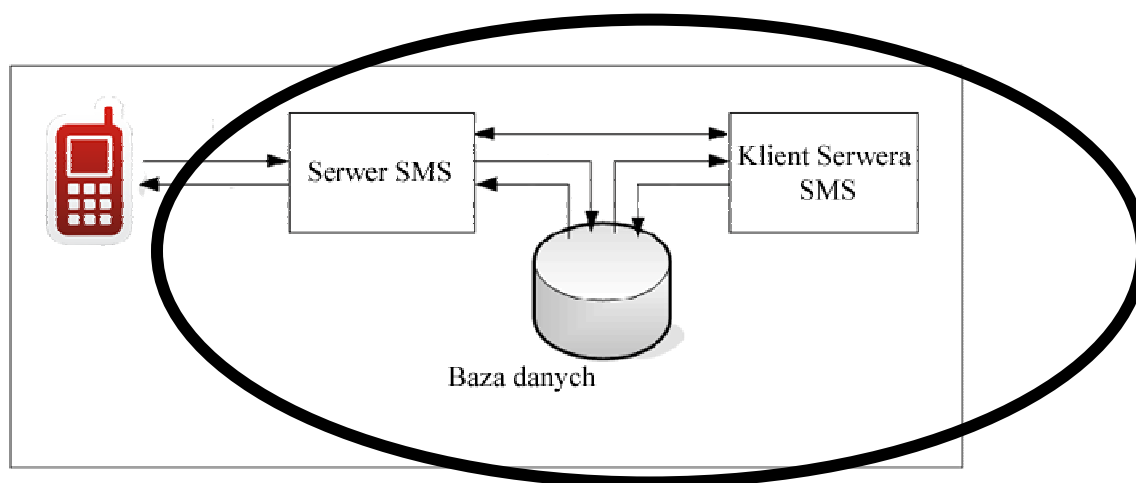
Na dobór technologii wykonania aplikacji klienckiej miały wpływ postawione wymagania oraz własne doświadczenie. Do utworzenia aplikacji webowej został

wykorzystany obiektowy język programowania PHP, warto zaznaczyć że język ten charakteryzuje się brakiem skomplikowanych typów i struktur danych, jest interpretowany tzn. wszelkie zmiany wprowadzone w kodzie są widoczne na serwerze po zapisaniu pliku, bez potrzeby kompilacji całego projektu.

5.2.3 Wymagania systemowe i sprzętowe

Można wyróżnić dwa rodzaje wymagań aplikacji. Dla serwera gdzie będzie się znajdowała aplikacja, oraz dla klienta końcowego. Do działania webaplikacji wymagany jest serwer WWW. Może nim być np. popularny Apache, Nginx lub lighthttp. W przypadku projektu został użyty ten pierwszy. Kolejnymi elementami jest interpreter kodu PHP oraz ta sama co w przypadku SMSS – baza danych. Wszystkie te i inne elementy wchodzą w skład pakietu oprogramowania XAMPP który po instalacji tworzy gotowe i w pełni skonfigurowane środowisko. Aby korzystać z aplikacji SMSC, użytkownik musi posiadać:

- dostęp do serwera na którym znajduje się aplikacja kliencka,
- przeglądarkę internetową (Internet Explorer, Mozilla Firefox, Opera itp.).



Rysunek 9 Zakres komunikacji aplikacji Klient SMS

5.2.4 Wykorzystanie zewnętrznych bibliotek programistycznych

Do realizacji projektu aplikacji klienckiej zostały wykorzystane biblioteki:

- jQuery – jest to biblioteka programistyczna dla języka JavaScript, posiadająca możliwość tworzenia efektownych animacji, modyfikowania treści stron WWW czy wykonywać zapytania AJAX,

- MDB2 – jest to biblioteka wchodząca w skład pakietu PEAR (PHP Extension and Application Repository) wykorzystywana do komunikacji z bazą danych.

5.2.5 Zaimplementowane funkcjonalności

5.2.5.1 Mechanizm zarządzania użytkownikami

Aby zapewnić rozliczalność, utworzony został system zarządzania użytkownikami . Możliwe jest tworzenie indywidualnych kont użytkowników, oraz przypisywanie do nich takich parametrów jak:

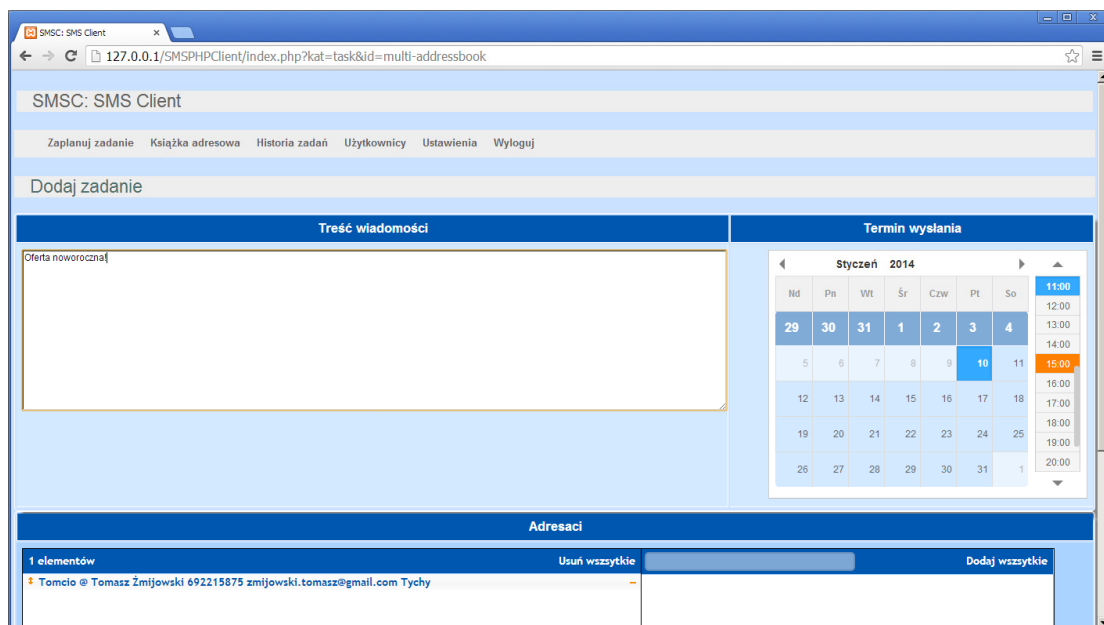
- login,
- hasło,
- imię,
- nazwisko,
- dział,
- blokada konta,
- poziom dostępu.

Poziom dostępu jest powiązany z priorytetowaniem zadań – im wyższy, tym w przypadku kolejki wiadomość zostanie szybciej wysłana. Poziom dostępu 100 określa dodatkowe funkcjonalności administratorskie jak:

- wspomniane zarządzanie użytkownikami,
- blokowanie numerów,
- dodawanie aktualności dla użytkowników,
- dostęp do całkowitej historii zadań.

5.2.5.2 Możliwość planowania wysyłki wiadomości

W ramach definiowania zadań, użytkownik ma możliwość zdefiniowania terminu wysłania wiadomości poprzez wybranie interesującego go dnia i godziny.



Rysunek 10 Planowanie wysyłki zadań

Po zatwierdzeniu zadania, tworzony jest w bazie danych rekord zawierający znacznik czasu po którym wiadomość będzie miała być wysłana. Zaletą takiego rozwiązania jest możliwość definiowania różnego rodzaju powiadomień które zostaną zrealizowane automatycznie w późniejszym czasie.

5.2.5.3 Nadawanie priorytetów wiadomościom

Z uwagi na to że system powiadomień może być stosowany do wysłania wiadomości o różnym priorytecie, zaimplementowana została możliwość nadawania priorytetów. W projekcie, priorytet będzie odzwierciedleniem poziomu dostępu (Access level). W przypadku gdy w kolejce zadań jest dużo wiadomości zdefiniowanych przez użytkownika o poziomie dostępu 1, a pojawi się potrzeba wysłania pilnego monitu np. niedostępności usług sieciowych przez konto a poziomie dostępu 100 to te zadanie zostanie obsłużone w pierwszej kolejności.

5.2.5.4 Książka adresowa

Książka adresowa to nic innego jak zbiór kontaktów zapisanych w bazie danych. W ramach każdego kontaktu istnieje możliwość przypisania podstawowych informacji o kontakcie:

- Imię, nazwisko, pseudonim,
- Numer telefonu,
- Adres zamieszkania,
- Email.

Kontakty - książka adresowa						
Numer	Pseudonim	Imię	Nazwisko	Adres	E-mail	Edytuj
692215875	Tomcio	Tomasz	Żmijowski	Tychy	zmijowski.tomasz@gmail.com	Edytuj
65256985	Mały	Janek	Mały	Katowice, Sikorskiego 34	maly@janek.com	Edytuj

Copyright © 2014 Tomasz Żmijowski

Rysunek 11 Książka adresowa – widok panelu zarządzania

Dzięki książce adresowej możliwe jest szybsze definiowanie odbiorców wiadomości poprzez ich wybieranie z zintegrowanej listy kontaktów.

		Dodaj wszystkie
Tomcio @ Tomasz Żmijowski	692215875 zmijowski.tomasz@gmail.com Tychy	+
Mały @ Janek Mały	65256985 maly@janek.com Katowice, Sikorskiego 34	+

Rysunek 12 Książka adresowa – widok panelu wyboru kontaktów

5.2.5.5 Historia zrealizowanych zadań

Historia wykonanych zadań umożliwia podgląd zadań wykonanych w przeszłości. W zależności od typu konta użytkownika – można przeglądać jedynie własną historię, lub w przypadku profilu administratora – historię wszystkich użytkowników systemu.

Tego typu rozwiązanie zapewnia, że użytkownicy mają pewność wykonania zadania, a podgląd historii całościowej z poziomu konta administratora pozwala na wykrywanie nadużyć w korzystaniu z firmowego systemu.

Historia zadań				
<div> <div>Moje -> admin</div> <div>Moje -> admin</div> <div> Wszystkie tom @ Tomasz Żmijowski IT admin @ Jan Kowalski Administracja Robert Krawczyk Obsługa Klienta Anna Mucha Obsługa Klienta Zadania w trakcie realizacji </div> </div> <div>Wyszukaj</div>				
Numer	Wiadomość	Termin wysłania	Obsługiwane przez	Anuluj
692215875	Nowe zadanie	2014-01-10 12:46:35	COM6	<input type="checkbox"/>
				Anuluj
Zadania wykonane				
Numer	Wiadomość	Termin wysłania	Obsłużono przez	Usun
692215875	hej	2013-12-28 13:42:59	COM6	<input type="checkbox"/>
692215875	asd	2013-12-28 14:06:18	COM6	<input type="checkbox"/>
692215875	asdsddd	2013-12-28 14:09:30	COM6	<input type="checkbox"/>
692215875	asd	2013-12-28 14:10:10	COM6	<input type="checkbox"/>
692215875	Kiedy spotkanie?	2013-12-28 14:13:25	COM6	<input type="checkbox"/>
692215875	Co?	2013-12-28 14:21:00	COM6	<input type="checkbox"/>
692215875	Spotkanie jutro o 17:00. Pozdrawiam!	2013-12-28 15:04:27	COM6	<input type="checkbox"/>

Rysunek 13 Historia zadań

5.2.5.6 Mechanizm blokowania numerów

Możliwość blokowania numerów, w tym numerów typu premium (koszty wysłania wiadomości dochodzące do kilkudziesięciu złotych za wiadomość), sprawia że użytkownicy systemu nie będą w stanie wysłać wiadomości na te numery.

Dodaj zadanie - weryfikacja	
Podsumowanie	
Treść wiadomości	start
Termin wysłania	2014/01/10 12:54:09
Odbiorcy	79523 <=> numer zablokowany
Wykonaj	
Copyright © 2014 Tomasz Żmijowski	

Rysunek 14 Weryfikacja numerów

Funkcja ta realizowana jest na poziomie aplikacji klienckiej. Podczas dodawania zadań do bazy numery odbiorców są porównywane z numerami zablokowanymi. Dodawania numerów zakazanych możliwe jest z poziomu konta administratora.

Zablokuj numer

Numer	Komentarz	Akcja
<input type="text" value="56235682"/>	<input type="text" value="Zablokowany klient - Nowak"/>	<input type="button" value="Prześlij"/>

Zablokowane numery

ID	Numer	Komentarz	Modyfikuj	Usuń
4	<input type="text" value="79XXX"/>	<input type="text" value="k1"/>	<input type="button" value="Aktualizuj"/>	<input type="button" value="Usuń"/>
6	<input type="text" value="78XXX"/>	<input type="text" value="k2"/>	<input type="button" value="Aktualizuj"/>	<input type="button" value="Usuń"/>
7	<input type="text" value="74XXX"/>	<input type="text" value="k3"/>	<input type="button" value="Aktualizuj"/>	<input type="button" value="Usuń"/>
9	<input type="text" value="71XXX"/>	<input type="text" value="k4"/>	<input type="button" value="Aktualizuj"/>	<input type="button" value="Usuń"/>

Copyright © 2014 Tomasz Żmijowski

Rysunek 15 Zarządzanie numerami zakazanymi

5.2.5.7 Aktualności systemowe

Funkcja aktualności systemowych pozwala na informowanie użytkowników systemu SMSC, poprzez wyświetlanie wiadomości na stronie głównej aplikacji.

Aktualności	
2013-12-27 22:41:10	Zablokowano numery premium - 79XXX, 78XXX, 74XXX, 71XXX
2013-12-22 17:41:20	W dniu dzisiejszym została uruchomiona usługa w wersji BETA. Wszelkie uwagi proszę kierować do administratora systemu.

Rysunek 16 Podgląd aktualności na stronie głównej

Dodaj aktualności

Treść	Akcja
<div></div>	<input type="button" value="Prześlij"/>

Modyfikuj aktualności

ID	Wiadomość	Data wiadomości	Modyfikuj	Usuń
1	<input type="text" value="W dniu dzisiejszym została uruchom"/>	2013-12-22 17:41:20	<input type="button" value="Aktualizuj"/>	<input type="button" value="Usuń"/>
5	<input type="text" value="Zablokowano numery premium - 7"/>	2013-12-27 22:41:10	<input type="button" value="Aktualizuj"/>	<input type="button" value="Usuń"/>

Rysunek 17 Zarządzanie aktualnościami

5.2.6 Wykorzystanie bazy danych


Baza danych wykorzystywana jest zarówno przez aplikację kliencką, jak i aplikację serwer SMS.

Celem utworzenie wspólnej bazy danych było przechowywanie:

- zadań przeznaczonych do wykonania przez SMSS,
- historii wysłanych wiadomości,
- informacji o użytkownikach systemu,
- informacji o kontaktach,
- listy zablokowanych numerów,
- aktualności dla użytkowników.

5.2.6.1 Przetwarzanie zadań

Dodając zadanie przez klienta systemu, w tabeli **tasks** pojawia się nowy rekord. SMSS po odpytaniu bazy danych i stwierdzeniu, że znajduje się w niej nowe zadanie, rezerwuje je, modyfikując pole **selected_by** nazwą aktualnie uruchomionej instancji programu.

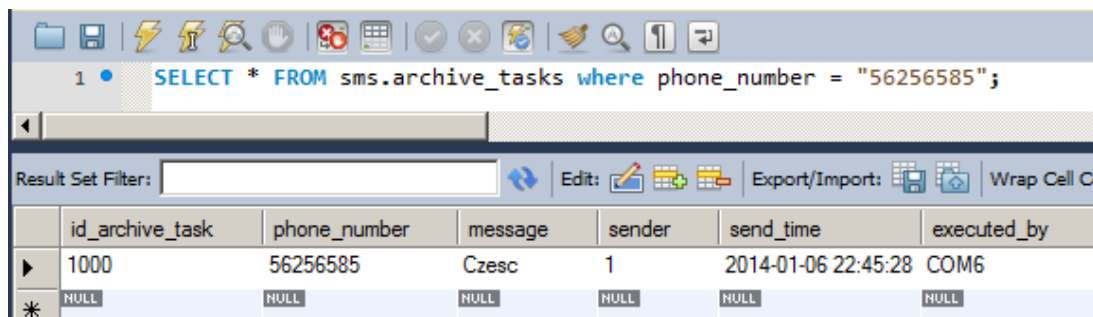


The screenshot shows a database query result window. At the top, there is a 'Result Set Filter' field and buttons for 'Edit', 'Export/Import', and 'Wrap Cell'. Below this is a table with the following columns: **id_task**, **phone_number**, **message**, **sender**, **send_time**, and **selected_by**. The first row contains the values: 1000, 56256585, Czesć, 1, 2014-01-06 22:45:28, and COM6. Below this row is a row with asterisks (*) and NULL values for all columns.

	id_task	phone_number	message	sender	send_time	selected_by
▶	1000	56256585	Czesć	1	2014-01-06 22:45:28	COM6
*	NULL	NULL	NULL	NULL	NULL	NULL

Rysunek 18 Rezerwacja zadania przez SMSS

Po wysłaniu wiadomości, SMSS usuwa zadanie z tabeli **tasks**, a zaimplementowany trigger (wyzwalacz), tworzy nowy rekord w tabeli **archive_tasks**.

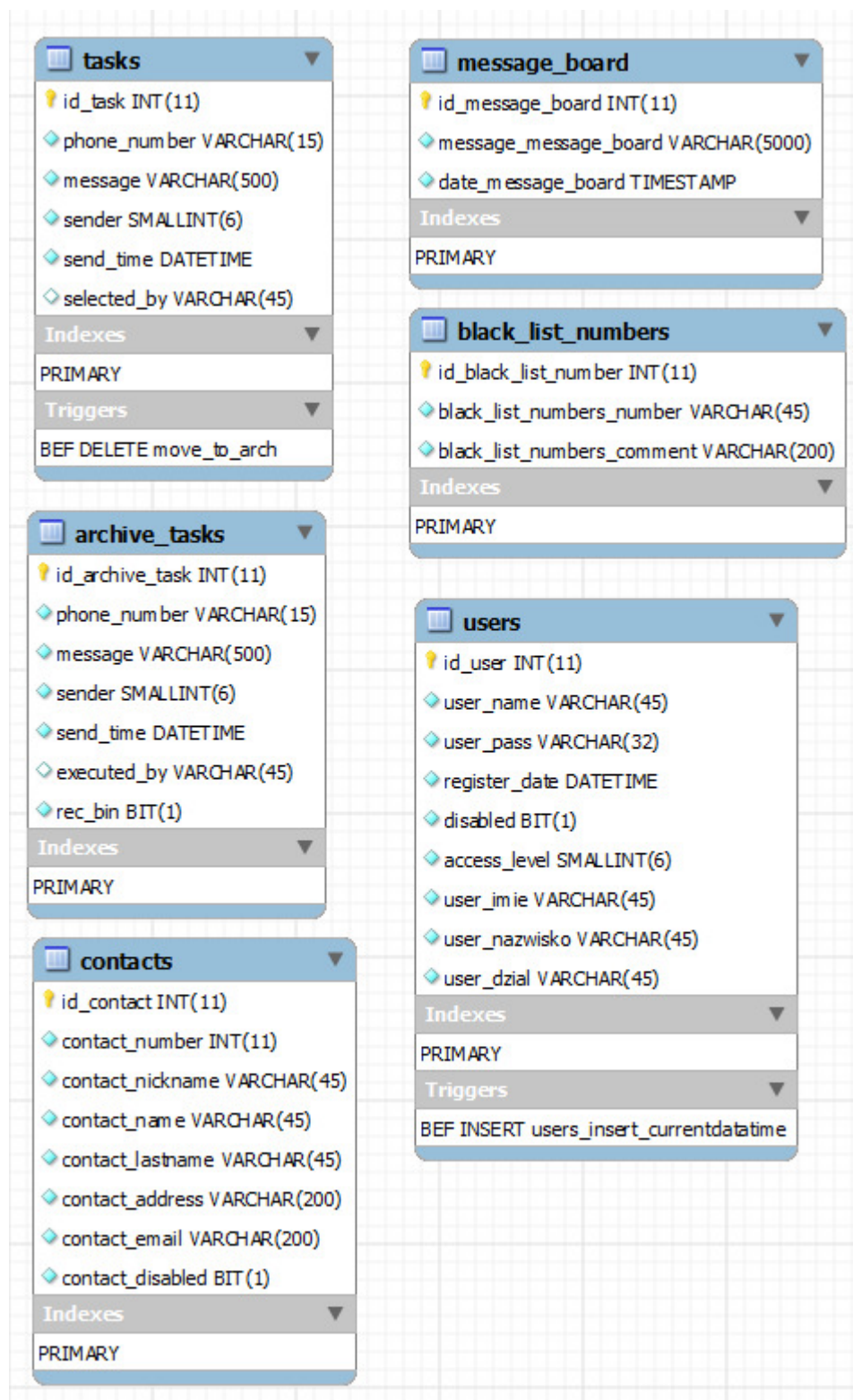


The screenshot shows a database query result window. At the top, there is a SQL query: `SELECT * FROM sms.archive_tasks where phone_number = "56256585";`. Below this is a table with the following columns: **id_archive_task**, **phone_number**, **message**, **sender**, **send_time**, and **executed_by**. The first row contains the values: 1000, 56256585, Czesć, 1, 2014-01-06 22:45:28, and COM6. Below this row is a row with asterisks (*) and NULL values for all columns.

	id_archive_task	phone_number	message	sender	send_time	executed_by
▶	1000	56256585	Czesć	1	2014-01-06 22:45:28	COM6
*	NULL	NULL	NULL	NULL	NULL	NULL

Rysunek 19 Wiadomość archiwalna

5.2.6.3 Schemat bazy danych



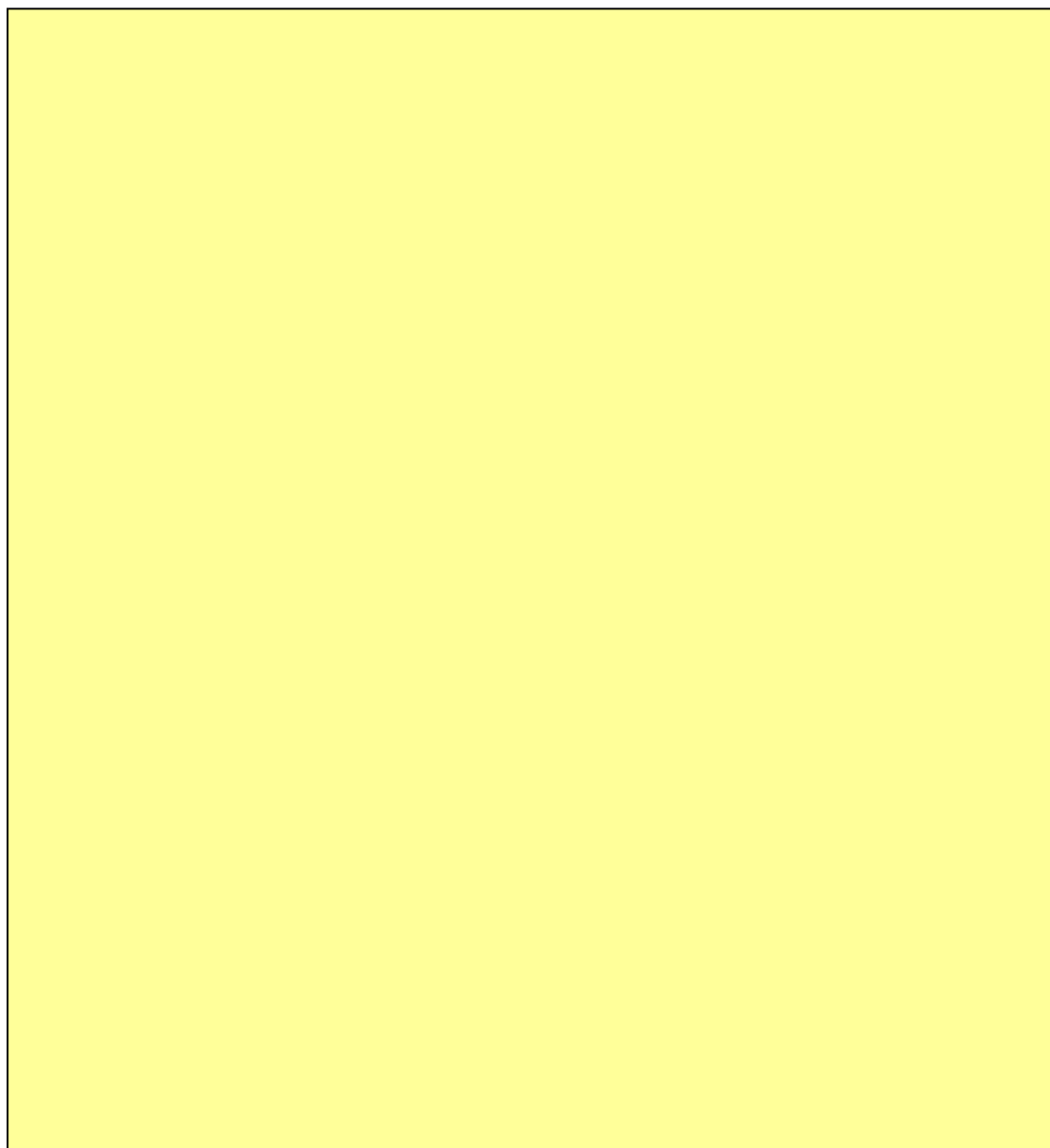
Rysunek 20 Schemat bazy danych

5.2.7 Bezpośrednie odwoływanie się do SMSC poprzez Socket

W związku z zaimplementowaniem bezpośredniego odwołania się do aplikacji serwer SMS poprzez Socket możliwa jest komunikacja typu klient-serwer [3].

Aby odwołać się do SMSS należy posiadać:

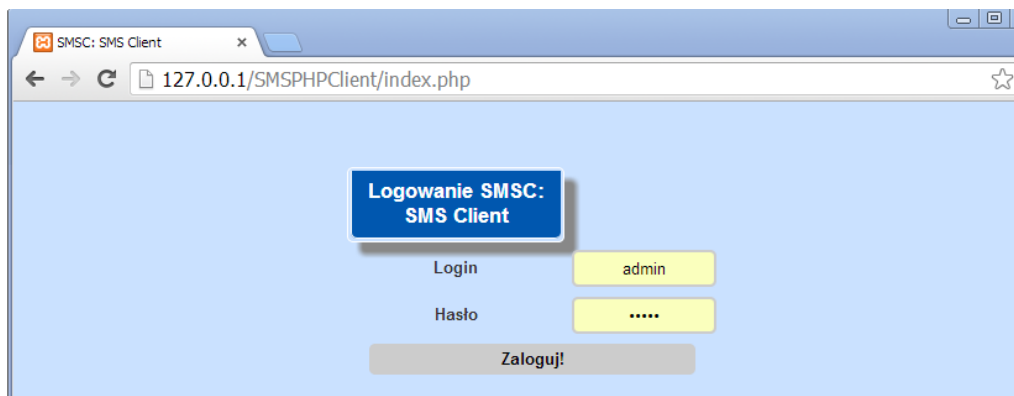
- Adres IP maszyny na którym uruchomiona jest aplikacja,
- Port na którym aplikacja działa,
- Poświadczenia zdefiniowane w konfiguracji aplikacji: użytkownika i hasło.



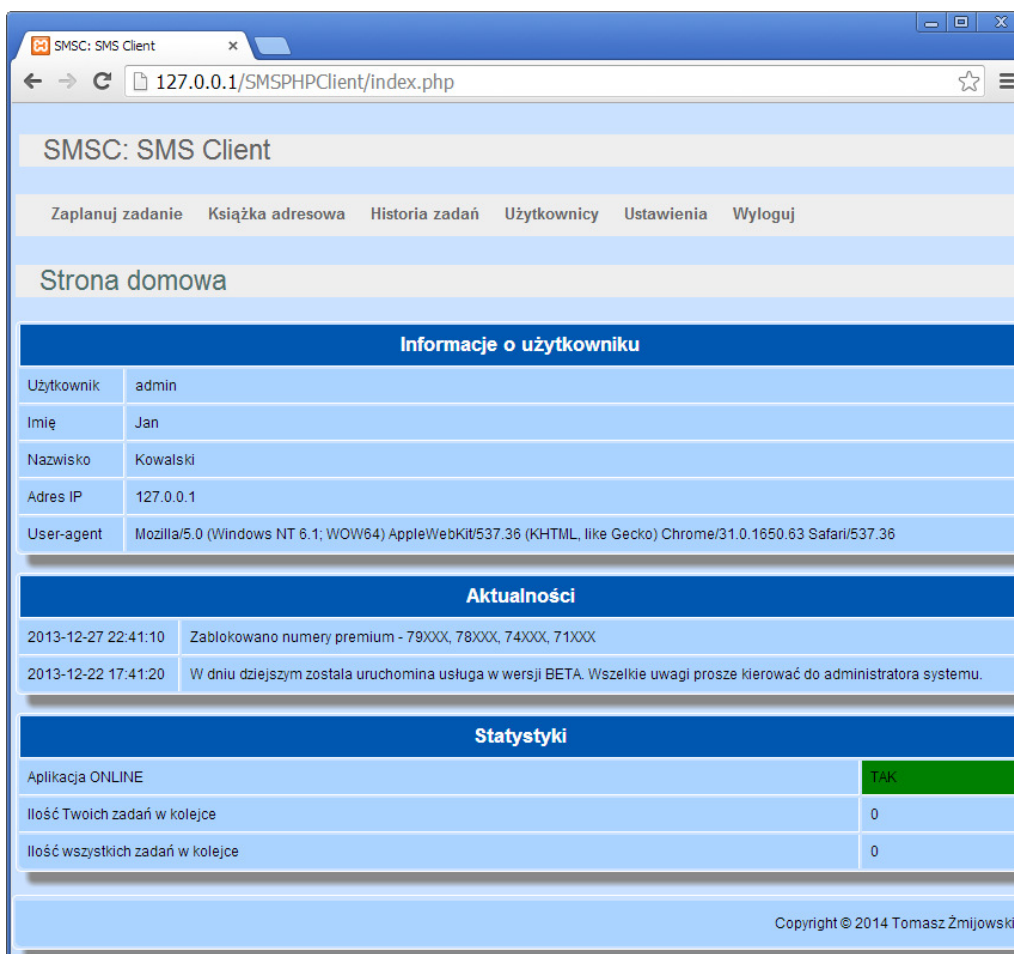
Listing kodu 3 Skrypt połączenia poprzez Socket

Skrypt posiada funkcje odczytu i zapisu danych z wykorzystaniem połączenia Socket. W zmiennej **\$pakiet** definiowane jest zapytanie do serwera, na podstawie którego aplikacja kliencka dostanie odpowiedź. W przypadku **getAppConfig** zostanie zwrócona konfiguracja serwera.

5.2.8 Podgląd działania aplikacji



Rysunek 21 Strona logowania



Rysunek 22 Strona główna aplikacji

6 Podsumowanie i wnioski

Projekt zrealizowany w ramach pracy inżynierskiej, jest system powiadomień SMS, którego głównym zadaniem jest wysyłanie krótkich wiadomości tekstowych do zdefiniowanych odbiorców. System taki znajdzie swoje zastosowanie zarówno od strony biznesowej przedsiębiorstwa – wysyłanie komunikatów do klientów, oraz może być wykorzystany wewnątrz przedsiębiorstwa do informowania pracowników. Zastosowanie popularnych technologii JAVA/PHP, umożliwiło stworzenie systemu mogącego być w prosty sposób implementowanym, oraz w razie potrzeb, dostosowywanym do wymagań biznesowych.

Biorąc pod uwagę czas, nakład pracy poświęcony na realizację projektu, można wyciągnąć następujące wnioski:

- dzięki zastosowaniu popularnych i otwartych technologii, istnieje możliwość łatwej modyfikacji i dostosowania pod specyfikę odbiorcy,
- SMSS jest w stanie funkcjonować zarówno w systemach z rodziny Windows jak i Linux czy Solaris, dzięki wykorzystaniu popularnego języka JAVA,
- aplikacja kliencka pozwala na intuicyjne korzystanie, definiowanie zadań i w konsekwencji wysłanie wiadomości SMS, bez konieczności znajomości zagadnień związanych z realizacją tego procesu,
- możliwość korzystania z aplikacji klienckiej przez przeglądarkę internetową, minimalizuje problemy związane z kompatybilnością oprogramowania.

Systemy powiadomień SMS są coraz częściej wykorzystywane przez różnego rodzaju przedsiębiorstwa usługowe czy handlowe. Nie raz mogłem uświadczyc otrzymywania wiadomości SMS przy korzystaniu z bankowości elektronicznej, zakupów online czy informacji, nt. godziny dostarczenia paczki przez kuriera. Rozwiązanie to zostało przyjęte przez firmy dlatego, że informacja trafia bezpośrednio na telefon komórkowy, który mamy zawsze pod ręką, a odbieranie komunikatu odbywa się w bardzo szybki i przyswajalny sposób.

Spis literatury

- [1] Herbert Schildt.: Java: Kompendium programisty, wydanie VIII, Helion, Gliwice 2012,
- [2] Jacek Ross.: PHP i HTML: Tworzenie dynamicznych stron WWW, Helion, Gliwice 2010,
- [3] Luke Welling, Laura Thomson.: PHP i MySQL. Tworzenie stron WWW. Wydanie drugie. Vademecum profesjonalisty, Helion, Gliwice 2003,
- [4] 3GPP TS 27.007 version 11.8.0 Release 11, AT command set for User Equipment (UE)
- [5] 3GPP TS 23.040 version 11.5.0 Release, Technical realization of the Short Message Service (SMS)
- [6] AT Commands, Nokia, 14.08.2008,
http://developer.nokia.com/Community/Wiki/AT_Commands,
- [7] Alexander Traud, 2014, <http://www.traud.de/gsm/>
- [8] Developer's Home, 2014, <http://www.developershome.com/sms/>,
- [9] Keane Jarki, 4.02.2009, <http://users.frii.com/jarvi/rxtx/>,
- [10] Elektronika Praktyczna 8/2002, <http://ep.com.pl/files/8073.pdf>.

Spis rysunków, tabel i kodów

Tabele

Tabela 1 Zestawienie ilości znaków w wiadomościach SMS	7
Tabela 2 Znaczenie poszczególnych oktetów wiadomości zapisanej w formacie PDU	8
Tabela 3 Zestawienie przykładowych komend AT (Basic AT Command Set)	10
Tabela 4 Zestawienie przykładowych komend AT (Extended AT Command Set)	10

Rysunki

Rysunek 1 Przykład wykorzystania komend AT programem Tera Term	12
Rysunek 2 Zakres komunikacji aplikacji Serwer SMS	17
Rysunek 3 Schemat obsługi zadań	18
Rysunek 4 Podgląd dostępnych portów COM - Windows	19
Rysunek 5 Podgląd dostępnych portów COM - Linux	19
Rysunek 6 Log uruchamiania się aplikacji	22
Rysunek 7 Log wysyłania wiadomości	23
Rysunek 8 Log komunikacji poprzez Socket	23
Rysunek 9 Zakres komunikacji aplikacji Klient SMS	24
Rysunek 10 Planowanie wysyłki zadań	26
Rysunek 11 Książka adresowa – widok panelu zarządzania	27
Rysunek 12 Książka adresowa – widok panelu wyboru kontaktów	27
Rysunek 13 Historia zadań	28
Rysunek 14 Weryfikacja numerów	28
Rysunek 15 Zarządzanie numerami zakazanymi	29
Rysunek 16 Podgląd aktualności na stronie głównej	29
Rysunek 17 Zarządzanie aktualnościami	29
Rysunek 18 Rezerwacja zadania przez SMSS	30
Rysunek 19 Wiadomość archiwalna	30
Rysunek 20 Schemat bazy danych	31
Rysunek 21 Strona logowania	33
Rysunek 22 Strona główna aplikacji	33

Kody

Listing kodu 1 Wykrywanie dostępnych portów COM	20
Listing kodu 2 Tworzenie komendy wysyłającej wiadomość SMS	21
Listing kodu 3 Skrypt połączenia poprzez Socket	32