SKRYPT DO LABORATORIUM

WYMIANA I SKŁADOWANIE DANYCH MULTIMEDIALNYCH

ĆWICZENIE 1: Badanie warstwy prezentacji na przykładzie normy DICOM. Wymiana danych w sieciach medycznych – DICOM/ HL7. Systemy informacyjne PACS w medycynie – zagadnienia administracji.

autor:
dr inż. Jacek Rumiński

Gdańsk, 2010

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*
współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego.

1. Zagadnienia wstępne

1.1.Wymagania w odniesieniu do studenta

Osiągnięcie celów ćwiczenia będzie możliwe poprzez właściwe przygotowanie się studenta do zajęć. Będzie to możliwe poprzez:

- powtórzenie wiedzy nabytej w czasie wykładów,

- zapoznanie się z instrukcją do ćwiczenia ilustrującą podstawowe zagadnienia z zakresu tematyki ćwiczenia, a szczególnie formatu komunikatów w HL7 2.x i DICOM.

Ponadto wymagana jest od studenta:

- podstawowa umiejętność programowania,
- umiejętność twórczego myślenia.

1.2.Wymagania w odniesieniu do stanowiska laboratoryjnego

Stanowisko laboratoryjne powinno być wyposażone w komputer klasy PC z działającym oprogramowaniem przygotowanym specjalnie dla potrzeb realizacji ćwiczenia. Na komputerze dostępne powinna być zainstalowana instrukcja do ćwiczenia w wersji elektronicznej oraz pliki testowe.

Stosowane technologie i narzędzia w ćwiczeniu:

- oprogramowanie Osiris (do edycji atrybutów DICOM),
- oprogramowanie przesyłania obiektów IOD w DICOM (opisane w dokumentacji ćwiczenia, katalog *docs*),
- przykładowe programy (kody programów) do budowy i wymiany komunikatów HL7 (opisane w załącznikach),
- środowisko programistyczne NetBeans.

1.3.Cele ćwiczenia

Celem ćwiczenia jest praktyczna demonstracja wiedzy teoretycznej, przedstawionej w ramach wykładów. Realizując ćwiczenie laboratoryjne studenci zdobędą szereg nowych umiejętności w zakresie zastosowania norm HL7 i DICOM, w tym:

- interpretacji struktur danych DICOM,

- interpretacji komunikatów DICOM,

- konfigurowania oprogramowania wymiany danych w systemie PACS (na bazie DICOM),

- interpretacji struktur danych HL7,

- interpretacji komunikatów HL7,

- konfigurowania oprogramowania wymiany danych zgodnie z normą HL7.

1.4.Metody dydaktyczne

Student w pierwszej kolejności analizuje budowę komunikatów DICOM oraz HL7, a następnie przeprowadza ćwiczenie praktyczne w zakresie wymiany tych komunikatów pomiędzy przykładowymi węzłami systemu informacyjnego.

Wszystkie utworzone dokumenty (opinie i propozycje rozwiązań) i kody źródłowe programów wpisuje w dokumencie elektronicznym, który jednocześnie staje się sprawozdaniem z ćwiczenia laboratoryjnego.

Materiały wprowadzające i pomocnicze:

- DICOM – treść normy;
- HL7 – treść normy
- Programy przykładowe oraz literatura zgodnie z opisem w tym dokumencie.

1.5.Zasady oceniania

Ocenie podlegać będzie realizacja poszczególnych zadań w ramach danego ćwiczenia laboratoryjnego. Dodatkowo w czasie realizacji ćwiczenia przydzielone zostaną studentowi dodatkowe zadania. Student musi zrealizować ćwiczenie laboratoryjne. Dodatkowym warunkiem zaliczenia jest uzyskanie minimum 16 punktów na podstawie oceny sprawozdań/plików źródłowych.

Wykaz literatury podstawowej do ćwiczenia:

Wykaz literatury podstawowej:

1.	Skrypt z materiałami do przedmiotu „Wymiana i składowanie danych multimedialnych”
2.	Materiały do przedmiotu opracowane w formie edukacji na odległość, dostęp: http://uno.biomed.gda.pl
3.	NEMA, Norma DICOM, dostęp: http://medical.nema.org
4.	HL7, norma i dokumenty HL7, dostęp: http://www.hl7.org

Wykaz literatury uzupełniającej:

2. Przebieg ćwiczenia

L.p.	Zadanie
1.	Zapoznanie się z instrukcją laboratoryjną
2.	Interpretacja struktur danych DICOM – wizualizacja atrybutów i obrazów
3.	Wymiana danych pomiędzy węzłami w systemie PACS zgodnie z DICOM
4.	Interpretacja struktur danych HL7

5.	Wymiana danych pomiędzy węzłami systemu informatycznego w szpitalu, zgodnie z normą HL7

UWAGI!

1. PRZED przystąpieniem do ćwiczenia należy w katalogu d:\dydaktyka\studenci utworzyć własny katalog (imie_nazwisko), a następnie przekopiować tam zawartość katalogu d:\dydaktyka\wisd\cw1. Proszę pracować wyłącznie na wykonanej kopii plików.

3. Wprowadzenie do ćwiczenia

W czasie realizacji ćwiczenia realizowane będą zadania zgodnie z ich wcześniejszym opisem. Każde zadanie (z wyjątkiem zadania realizowanego przed przystąpieniem do ćwiczenia laboratoryjnego) opisano poniżej.

Ad 1. Zapoznanie się z instrukcją laboratoryjną (5 min.)

Po wprowadzeniu kierownika ćwiczenia uczestnicy zapoznają się ze stanowiskiem komputerowym, oprogramowaniem i dokumentami związanymi z ćwiczeniem.
Następnie utworzyć we wskazanym przez kierownika ćwiczenia katalogu własny podkatalog o nazwie zawierającej własne nazwisko. Do katalogu tego przegrać zawartość podkatalogu „DANE” (znajdującego się w folderze zawierającym instrukcję do ćwiczenia). W utworzonym podkatalogu należy przechowywać wszystkie wytworzone w czasie trwania ćwiczenia dokumenty i programy.

Ad 2. Interpretacja struktur danych DICOM – wizualizacja atrybutów i obrazów (20 min.)

Na podstawie rozdziału 3 normy DICOM (stanowiącej część dokumentacji ćwiczenia) zinterpretować treść zapisanych w plikach obiektów informacyjnych (../klient/ct.dcm). Wykorzystać do tego celu oprogramowanie OSIRIS oraz edytor programu Windows Commander.

Ad 3. Wymiana danych pomiędzy węzłami w systemie PACS zgodnie z DICOM (40 min.)

Wymiana i składowanie danych medycznych w sieciach bazujących na protokole TCP/IP.

Celem tej części ćwiczenia jest zapoznanie się z przesyłaniem i składowaniem obrazów zgodnie z normą DICOM.

W tej części ćwiczenia studenci wykorzystują dwa zestawy komputerowe wskazane przez prowadzącego, oraz oprogramowanie stanowiące efekt dwóch prac dyplomowych prowadzonych w Katedrze Inżynierii Biomedycznej, WETI, PG.

Na pierwszym stanowisku (wskazanym przez prowadzącego), np.:
IP: 153 . 19 . 51 . 114
należy uruchomić serwer (Storage SCP) składowania danych DICOM (../serwer/!run.bat). Spowoduje to wywołanie programu serwera oraz okna dialogowego nadzorującego pracę programu. Uruchomienie nasłuchu odbywa się poprzez wciśnięcie przycisku START. Plik konfiguracyjny (m.in. katalog roboczy do którego zapisywane są otrzymywane dane) oraz logi transmisji zawarte są w katalogu dicomworks/ (configdcm.txt).
Na tym samym stanowisku należy uruchomić również klienta obsługi składowania danych DICOM w bazie danych. W tym celu należy wywołać polecenie (../klientdb/!run.bat). Spowoduje to wywołanie aplikacji, oraz okna dialogowego z ustawieniami. W oknie tym należy zmienić jedynie treść pola hasło (podane przez prowadzącego). Następnie wciśnięcie przycisku „>” spowoduje uruchomienie wątka automatycznej aktualizacji bazy danych na podstawie otrzymywanych przez serwer DICOM danych. Sygnalizacja, oraz wczytanie nowych danych dostępne jest w dolnej części interfejsu graficznego aplikacji.

Wykorzystanie powyższych programów zademonstrowane jest poprzez uruchomienie na drugim stanowisku, np.:
IP: 153 . 19 . 51 . 119
klienta DICOM (Storage SCU). W tym celu najpierw należy zapoznać się z treścią pliku wsadowego ../klient/!run.bat. Następnie należy przygotować się na transmisję (aby swobodnie analizować komunikaty drukowane w oknach konsoli klienta i serwera DICOM) a następnie wywołać plik „!run.bat”.

Po prawidłowym przesłaniu pliku „ct.dcm”, klient bazy (stanowisko pierwsze) pokaże (po około 1 minucie – interwał aktualizacji) informacje o dostępie do nowych danych. Należy wczytać plik, a następnie zapoznać się z informacjami dostępnymi poprzez wywołanie przycisków „Edytuj” dla kolejnych struktur danych. Wywołując przeglądanie obrazu należy następnie przełączyć się między

oknami (Alt+TAB) w celu wyświetlenia okna z obrazem. Okno to zawiera dodatkowy przycisk, którego wywołanie spowoduje wydruk treści elementów pliku DICOM, zgodnie z formatem XML.

Po zapoznaniu się z powyższymi cechami należy w oknie klienta bazy usunąć wczytane dane (ikona X, przycisk Usuń, zatwierdzenie). Ćwiczenie jest zakończone, po powiadomieniu prowadzącego można wyłączyć programy i sprzęt komputerowy.

Ad 4. Interpretacja struktur danych HL7 (25 min.)

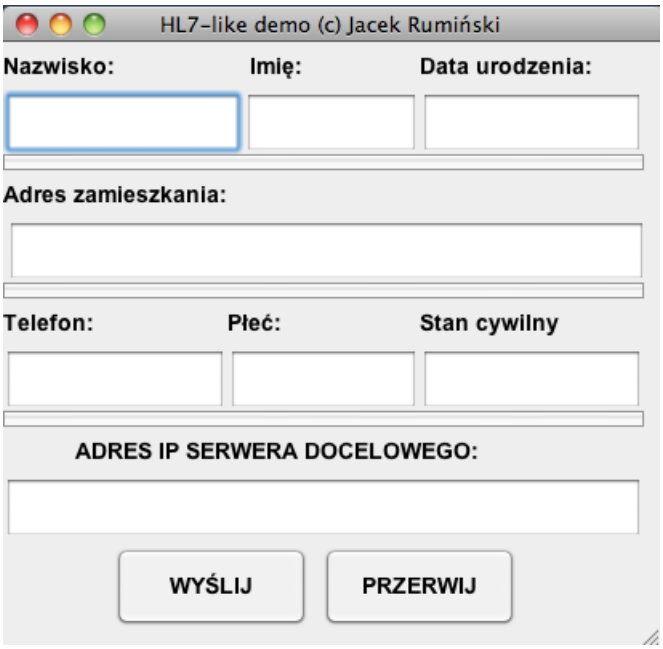
Na podstawie dokumentacji HL7 (stanowiącej część dokumentacji ćwiczenia) zinterpretować segmenty i dane komunikatu przedstawionego przez prowadzącego.

Ad. 5 Wymiana danych pomiędzy węzłami systemu informatycznego w szpitalu, zgodnie z normą HL7 (45 min.)

Wykorzystując oprogramowanie specjalnie utworzone dla potrzeb niniejszego ćwiczenia prześledzić proces wymiany komunikatu tekstowego pomiędzy węzłami sieci komputerowej. W tym celu najpierw należy uruchomić węzeł serwera (ilustracja odbiorcy komunikatu HL7) poprzez wywołanie pliku:
hl7serwer.bat.

Następnie należy uruchomić proces klienta poprzez uruchomienie programu klienta:
hl7client.bat.

Pojawi się wówczas prosty interfejs graficzny:



w którego formularzu wpisujemy przykładowe dane pacjenta. Dane te zostaną wykorzystane do utworzenia treści komunikatu HL7. Po wciśnięciu przycisku „Wyślij” komunikat HL7 (jako dokument tekstowy) zostanie przesłany do serwera.

Przeanalizować wyniki uzyskane w konsoli serwera.

Następnie w kodzie źródłowy dokonać modyfikacji budowy lub treści komunikatu HL7, skompilować program i ponownie uruchomić proces przesyłania komunikatu HL7.

Zastanowić się w jaki sposób możliwe jest odwzorowanie danych z hipotetycznej bazy danych adresata do komunikatu HL7 oraz jak można odwzorować sytuację odwrotną (HL7 -> BD).

4 Załączniki

W części tej zawarto kody programów oraz pliki przykładowe i instrukcje konfiguracji oprogramowania.

4.1 Kod źródłowy przykładu DICOM

Kody źródłowe oraz dokumentacja oprogramowania dostępna jest w katalogu docs, ćwiczenia laboratoryjnego. Ze względu na swoją złożoność nie jest kopiowana w tej instrukcji.

4.2. Kod źródłowy przykładu HL7

Oprogramowanie demonstracyjne składa się z 3 klasy:

- HL7message – repezentujące przykładowy komunikat,
- HL7serverDemo – klasa serwera, odbiorcy wiadomości HL7,
- HL7frameDemo – klasa klienta systemu.

```
/*
 * HL7message.java
 */

package org.cemte.jacek.hl7;
import java.util.*;

/**
  * @author  jwr
 */
public class HL7message {

    /* Przykładowy komunikat
   MSH|^~\&|HL7demo|IRR|HL7test|KEMiE|20000104121200||ADT^A29|WIAD123|P|2.3.1|||||8859/2<cr>
    EVN|A29|20000104121200<cr>
   PID|1||P00112^^^SPSK1G||Nowak^Jan||19601212|M|||Narutowicza 11/12^^Gdansk^^80-
952||(4858)3472645|||S<cr>
   PV1|1|E|||||Kowalski^Adam|||||||7<cr>
    *
     */
    public String MSH;
    public String EVN;
    public String PID;
    public String PV1;

    public String nazwisko="";
    public String imie="";
    public String data_ur="";
```

```
   public String adres="";
   public String plec="";
   public String stan_cyw="";
   public String telefon="";

   /** Creates a new instance of HL7message */
   public HL7message() {
       MSH="MSH|^~\\&|HL7demo|IRR|HL7test|KIB|"+new Date().toString()+
"||ADT^A29|WIAD123|P|2.3.1||||||8859/2\n";
      EVN="EVN|A29|"+new Date().toString()+"\n";
      PID="PID|1||P00112^^^SPSK1G||";
      PV1="PV1|1|E|||||Kowalski^Adam|||||||7\n";
   }

   public String generateMessage(){
      PID+=nazwisko+"^"+imie+"||"+data_ur+"|"+plec+"|||"+adres+"||"+telefon+"|||"+stan_cyw+"\n";
      return MSH+EVN+PID+PV1;

   }// generateMessage

}//class


/*
 * HL7serwerDemo.java
 */

package org.cemte.jacek.hl7;

import java.net.*;
import java.io.*;

/**
 * @author  jwr
 */
public class HL7serwerDemo {

   /** Creates a new instance of HL7serwerDemo */
   public HL7serwerDemo() {
   }
   public static void main(String[] args) {

   ServerSocket serwer;
   Socket     gniazdo;

   BufferedReader strumienWe;

   String echo;



    try {
     serwer = new ServerSocket(9009);
       while(true){ //główna pętla serwera
          try{
            while(true){ //główna pętla połączenia
```

```
            gniazdo = serwer.accept();  //przyjmuj po³¹czenia i stworz gniazdo
            System.out.println("Jest polaczenie");
            strumienWe = new BufferedReader(new InputStreamReader(gniazdo.getInputStream()));
            while((echo=strumienWe.readLine())!=null){
                 System.out.print("\nODEBRANO SEGMENT: "+echo.substring(0,3));
                 System.out.println("  O TREŚCI:");
                 System.out.println(echo);
            }
          }//od while
         }//od try
          catch (SocketException e){
             System.out.println("Zerwano polaczenie"); //klient zerwał połączenie
          }
          catch (IOException e) {
              System.err.println(e);
          }
      }//od while
     }// od try
     catch (IOException e) {
       System.err.println(e);
     } //catch
  }
}//class


/*
 * HL7frameDemo.java
 */

package org.cemte.jacek.hl7;

import java.util.*;
import java.awt.event.*;
import java.net.*;
import javax.swing.*;
import java.io.*;

/**
 * @author  jwr
 */
public class HL7frameDemo extends javax.swing.JFrame implements ActionListener {



   private String nazwisko;
   private String imie;
   private String data_ur;
   private String adres;
   private String plec;
   private String stan_cyw;


   /** Creates new form HL7frameDemo */
   public HL7frameDemo() {
      super("HL7-like demo (c) Jacek Rumiński");
```

```
    initComponents();

}

/** This method is called from within the constructor to
 * initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is
 * always regenerated by the Form Editor.
 */
private void initComponents() {
    jLabel1 = new javax.swing.JLabel();
    jTextField1 = new javax.swing.JTextField();
    jLabel2 = new javax.swing.JLabel();
    jTextField2 = new javax.swing.JTextField();
    jLabel3 = new javax.swing.JLabel();
    jTextField3 = new javax.swing.JTextField();
    jButton1 = new javax.swing.JButton();
    jLabel4 = new javax.swing.JLabel();
    jTextField4 = new javax.swing.JTextField();
    jButton2 = new javax.swing.JButton();
    jLabel5 = new javax.swing.JLabel();
    jTextField5 = new javax.swing.JTextField();
    jLabel6 = new javax.swing.JLabel();
    jTextField6 = new javax.swing.JTextField();
    jLabel7 = new javax.swing.JLabel();
    jTextField7 = new javax.swing.JTextField();
    jButton3 = new javax.swing.JButton();
    jButton4 = new javax.swing.JButton();
    jButton5 = new javax.swing.JButton();
    jLabel8 = new javax.swing.JLabel();
    jTextField8 = new javax.swing.JTextField();

    getContentPane().setLayout(null);

    addWindowListener(new java.awt.event.WindowAdapter() {
        public void windowClosing(java.awt.event.WindowEvent evt) {
            exitForm(evt);
        }
    });

    jLabel1.setFont(new java.awt.Font("Arial", 1, 14));
    jLabel1.setText("Nazwisko:");
    getContentPane().add(jLabel1);
    jLabel1.setBounds(0, 0, 150, 30);

    getContentPane().add(jTextField1);
    jTextField1.setBounds(0, 30, 150, 40);

    jLabel2.setFont(new java.awt.Font("Arial", 1, 14));
    jLabel2.setText("Imi\u0119:");
    getContentPane().add(jLabel2);
    jLabel2.setBounds(154, 0, 100, 30);

    getContentPane().add(jTextField2);
    jTextField2.setBounds(150, 30, 110, 40);
```

```
jLabel3.setFont(new java.awt.Font("Arial", 1, 14));
jLabel3.setText("Data urodzenia:");
getContentPane().add(jLabel3);
jLabel3.setBounds(260, 0, 140, 30);

getContentPane().add(jTextField3);
jTextField3.setBounds(260, 30, 140, 40);

jButton1.setText("_");
getContentPane().add(jButton1);
jButton1.setBounds(0, 70, 400, 10);

jLabel4.setFont(new java.awt.Font("Arial", 1, 14));
jLabel4.setText("Adres zamieszkania:");
getContentPane().add(jLabel4);
jLabel4.setBounds(0, 80, 400, 30);

getContentPane().add(jTextField4);
jTextField4.setBounds(2, 110, 400, 40);

jButton2.setText("_");
getContentPane().add(jButton2);
jButton2.setBounds(0, 150, 400, 10);

jLabel5.setFont(new java.awt.Font("Arial", 1, 14));
jLabel5.setText("Telefon:");
getContentPane().add(jLabel5);
jLabel5.setBounds(0, 160, 150, 30);

getContentPane().add(jTextField5);
jTextField5.setBounds(0, 190, 140, 40);

jLabel6.setFont(new java.awt.Font("Arial", 1, 14));
jLabel6.setText("P\u0142e\u0107:");
getContentPane().add(jLabel6);
jLabel6.setBounds(140, 160, 100, 30);

getContentPane().add(jTextField6);
jTextField6.setBounds(140, 190, 120, 40);

jLabel7.setFont(new java.awt.Font("Arial", 1, 14));
jLabel7.setText("Stan cywilny");
getContentPane().add(jLabel7);
jLabel7.setBounds(260, 160, 140, 30);

getContentPane().add(jTextField7);
jTextField7.setBounds(260, 190, 140, 40);


jButton5.setText("_");
getContentPane().add(jButton5);
jButton5.setBounds(0, 230, 400, 10);

jLabel8.setFont(new java.awt.Font("Arial", 1, 14));
```

```
    jLabel8.setText("ADRES IP SERWERA DOCELOWEGO:");
    getContentPane().add(jLabel8);
    jLabel8.setBounds(45, 240, 300, 30);

    getContentPane().add(jTextField8);
    jTextField8.setBounds(0, 270, 400, 40);

    jButton3.setFont(new java.awt.Font("Arial", 1, 14));
    jButton3.setText("WY\u015aLIJ");
    jButton3.addActionListener(this);
    getContentPane().add(jButton3);
    jButton3.setBounds(70, 315, 120, 50);

    jButton4.setFont(new java.awt.Font("Arial", 1, 14));
    jButton4.setText("PRZERWIJ");
    jButton4.addActionListener(new ActionListener(){
                        public void actionPerformed(ActionEvent ae){
                                System.exit(1);
                        }
                });
    getContentPane().add(jButton4);
    jButton4.setBounds(200, 315, 120, 50);

    pack();
  }

  /** Exit the Application */
  public void actionPerformed(ActionEvent ae){
        HL7message hlm = new HL7message();
        hlm.nazwisko= jTextField1.getText();
    hlm.imie= jTextField2.getText();
    hlm.data_ur= jTextField3.getText();
    hlm.adres= jTextField4.getText();
    hlm.telefon= jTextField5.getText();
    hlm.plec= jTextField6.getText();
    hlm.stan_cyw= jTextField7.getText();
    String msg = hlm.generateMessage();
    System.out.println(msg);

    String serwer=jTextField8.getText();
    try{
      InetAddress ia =InetAddress.getByName(serwer);
      wyslij(msg,serwer,9009);
    }catch (Exception e){
      JOptionPane.showMessageDialog(this,new String("Błąd adresu IP:"+e),new String("OSRZEŻENIE
!"),JOptionPane.WARNING_MESSAGE);
    }


  }

  public void wyslij(String msg, String adres, int port){
    try {
     Socket gniazdo = new Socket(adres, port);
     BufferedWriter strumien = new BufferedWriter(new OutputStreamWriter(gniazdo.getOutputStream()));
```

```
        strumien.write(msg);
        strumien.flush();
        JOptionPane.showMessageDialog(this,new String("Wysłano wiadomość "),new String("INFORMACJA
!"),JOptionPane.INFORMATION_MESSAGE);

      }
      catch (UnknownHostException e) {
        JOptionPane.showMessageDialog(this,new String("Błąd adresu IP:"+e),new String("OSRZEŻENIE
!"),JOptionPane.WARNING_MESSAGE);

      }
      catch (IOException e) {
        JOptionPane.showMessageDialog(this,new String("Błąd połączenia:"+e),new String("OSRZEŻENIE
!"),JOptionPane.WARNING_MESSAGE);
       //System.err.println(e);
       }
    }

    private void exitForm(java.awt.event.WindowEvent evt) {
      System.exit(0);
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
      HL7frameDemo hl=new HL7frameDemo();
      hl.setSize(410,400);
      hl.setVisible(true);//show();
    }

    // Variables declaration - do not modify
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JButton jButton3;
    private javax.swing.JButton jButton4;
     private javax.swing.JButton jButton5;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
     private javax.swing.JLabel jLabel8;
    private javax.swing.JTextField jTextField1;
    private javax.swing.JTextField jTextField2;
    private javax.swing.JTextField jTextField3;
    private javax.swing.JTextField jTextField4;
    private javax.swing.JTextField jTextField5;
    private javax.swing.JTextField jTextField6;
    private javax.swing.JTextField jTextField7;
    private javax.swing.JTextField jTextField8;
    // End of variables declaration
```

}//class