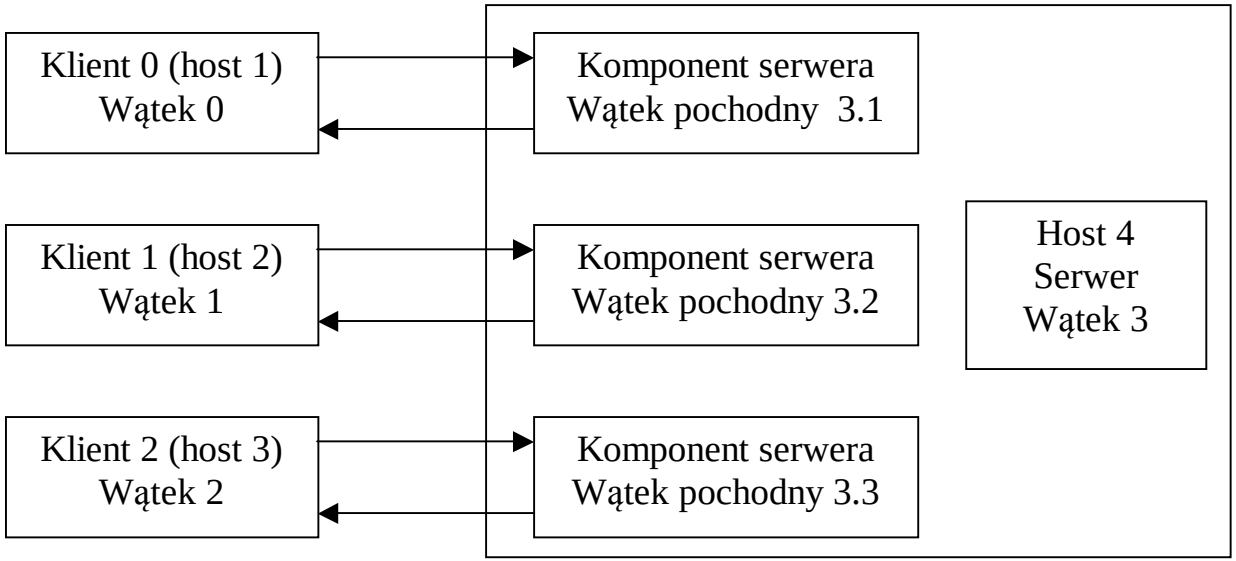
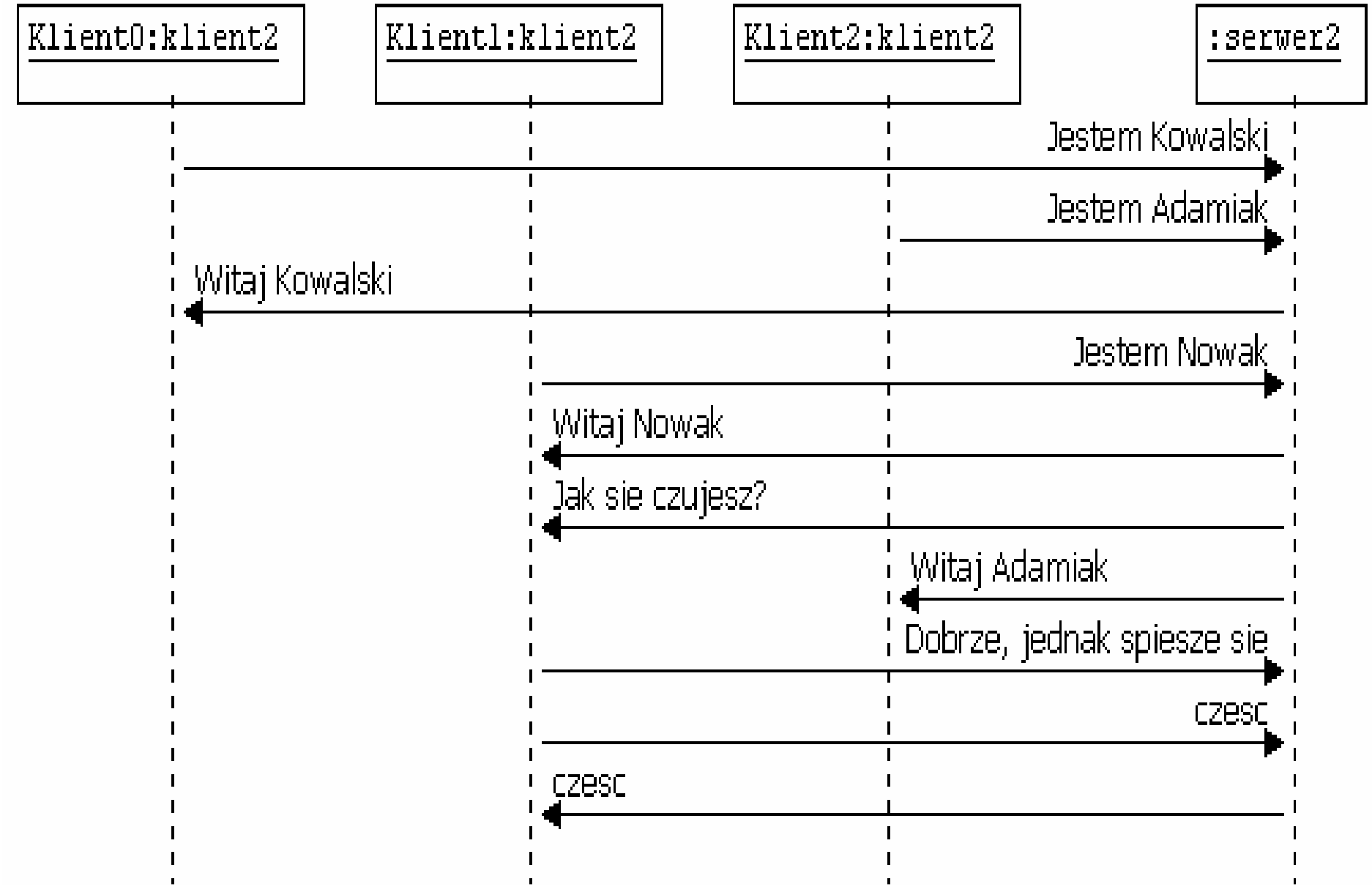
Aplikacja wielowątkowa – prosty komunikator

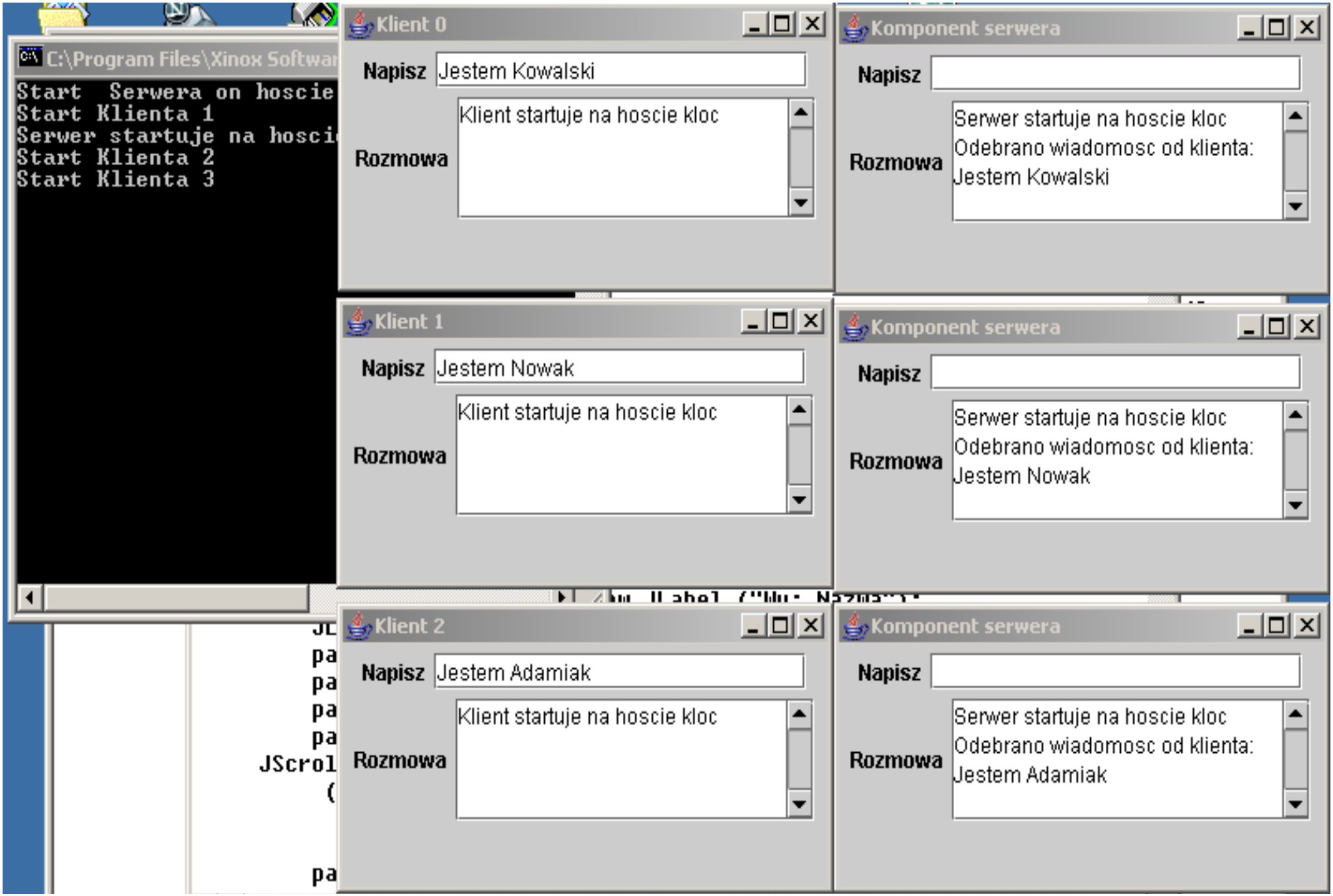


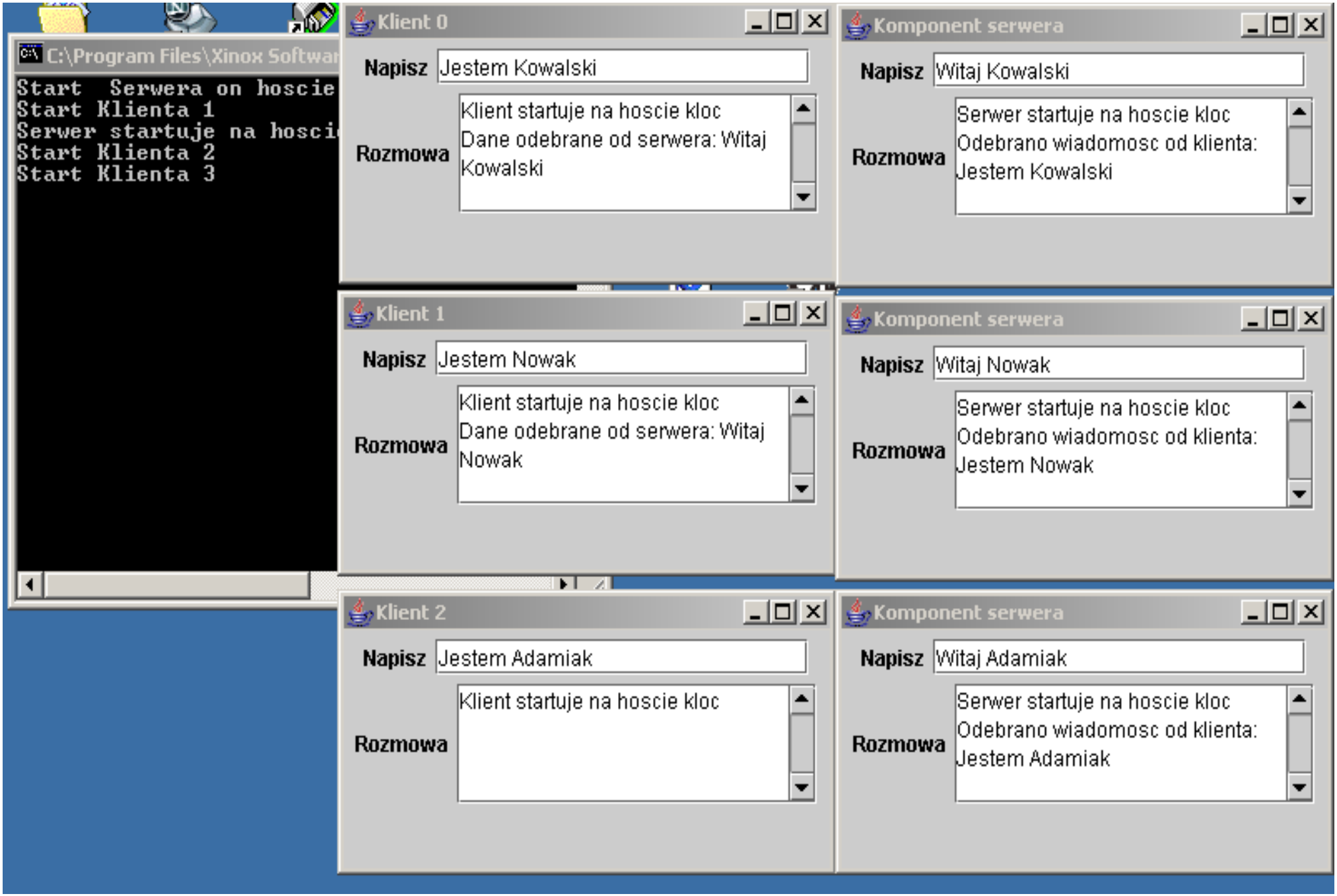
Klient: **private int** sPort *//port serwera*
private String host *//nazwa hosta serwera*
private Socket s *//gniazdo klienta do komunikacji z serwerem, który znajduje*
//się na porcie sPort i na komputerze host
private ObjectOutputStream output
private ObjectInputStream input

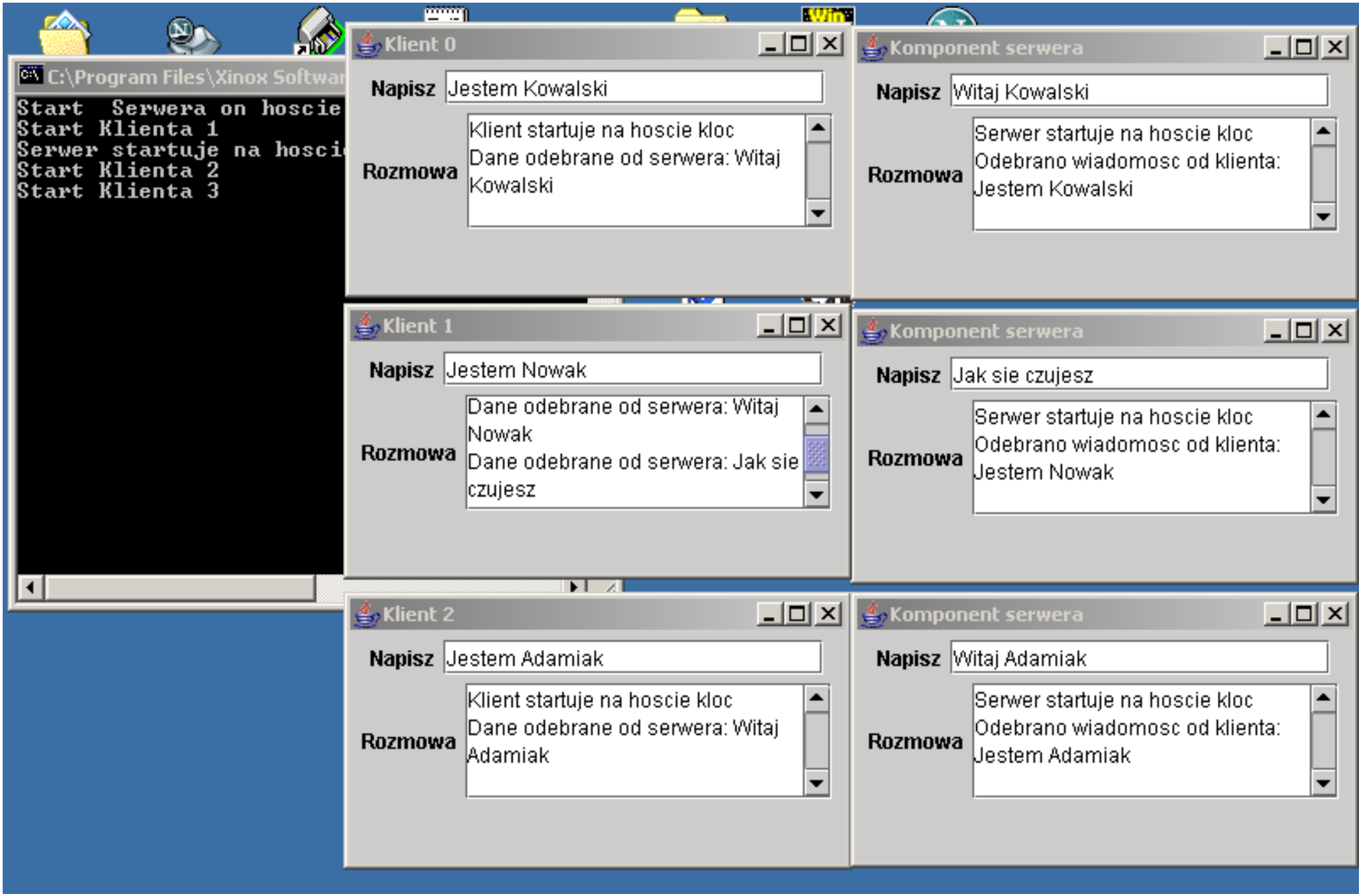
Serwer: **private int** sPort
private String host
private ServerSocket serwer *//gniazdo do wykrywania połączeń z klientem*

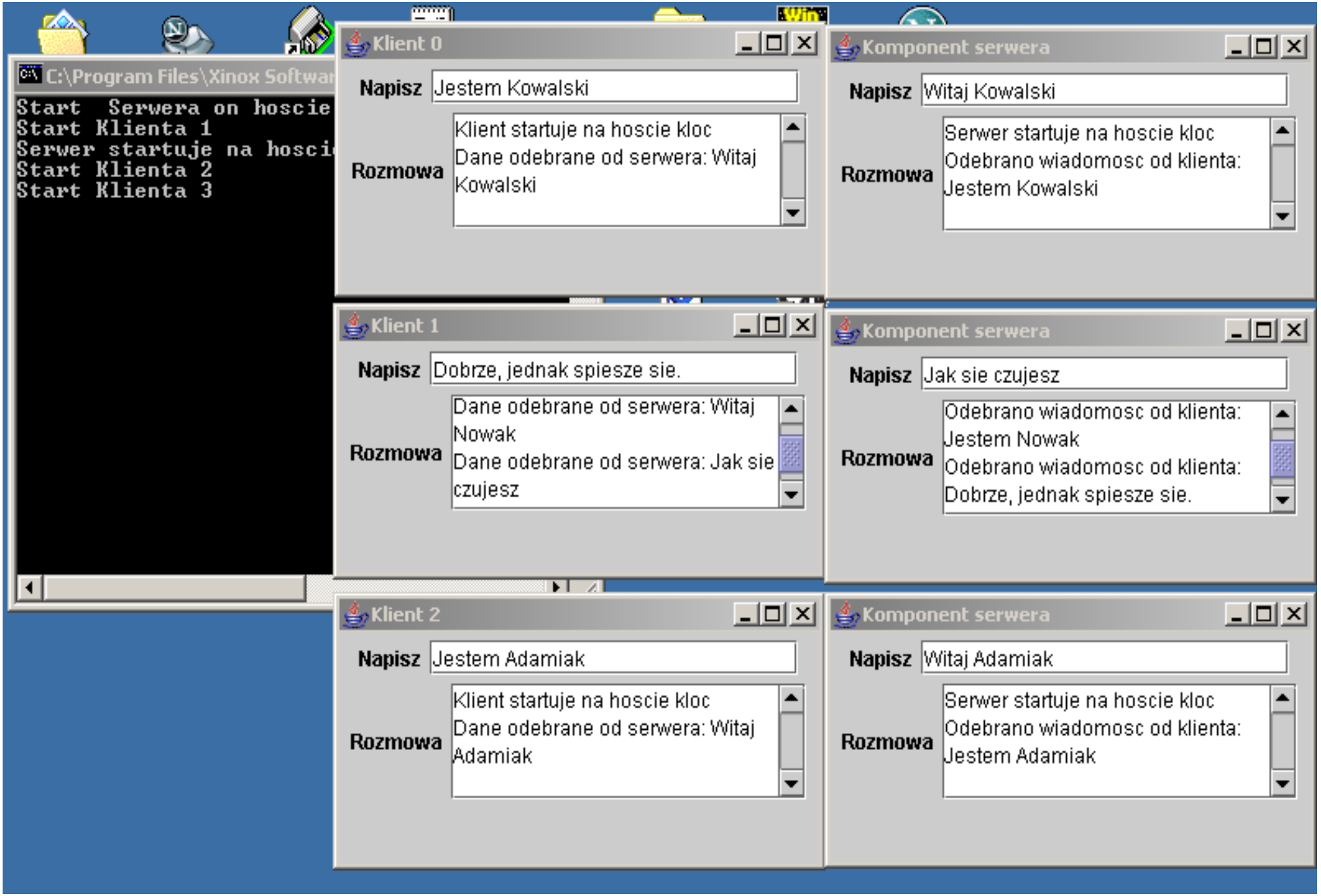
Komponent serwera: **private** Socket s *//gniazdo do komunikacji z klientem*
private ObjectOutputStream output
private ObjectInputStream input

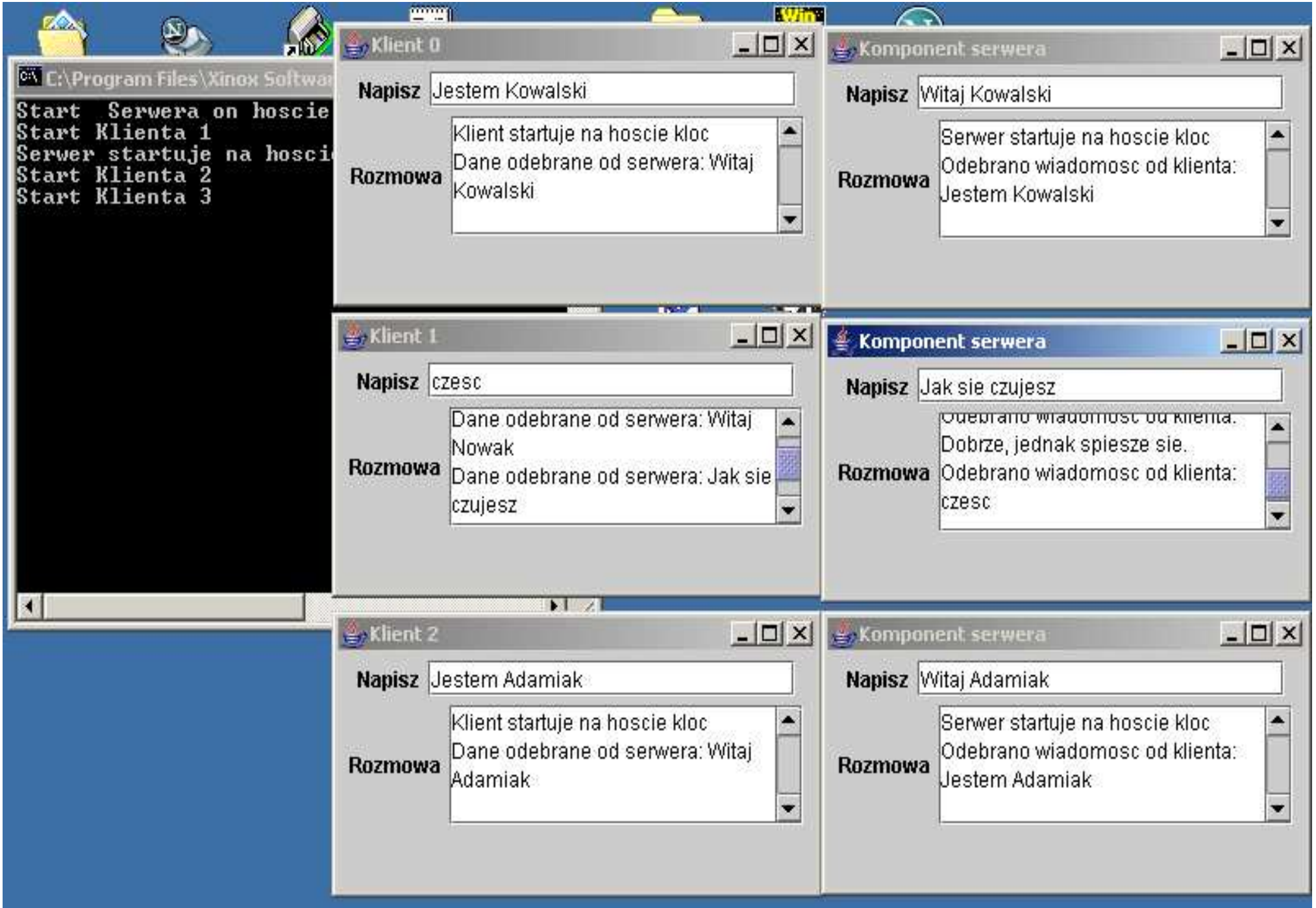

Klient0:klient2
Klient1:klient2
Klient2:klient2
:serwer2
Jestem Kowalski
Jestem Adamiak
Witaj Kowalski
Jestem Nowak
Witaj Nowak
Jak sie czujesz?
Witaj Adamiak
Dobrze, jednak spiesze sie
czesc
czesc

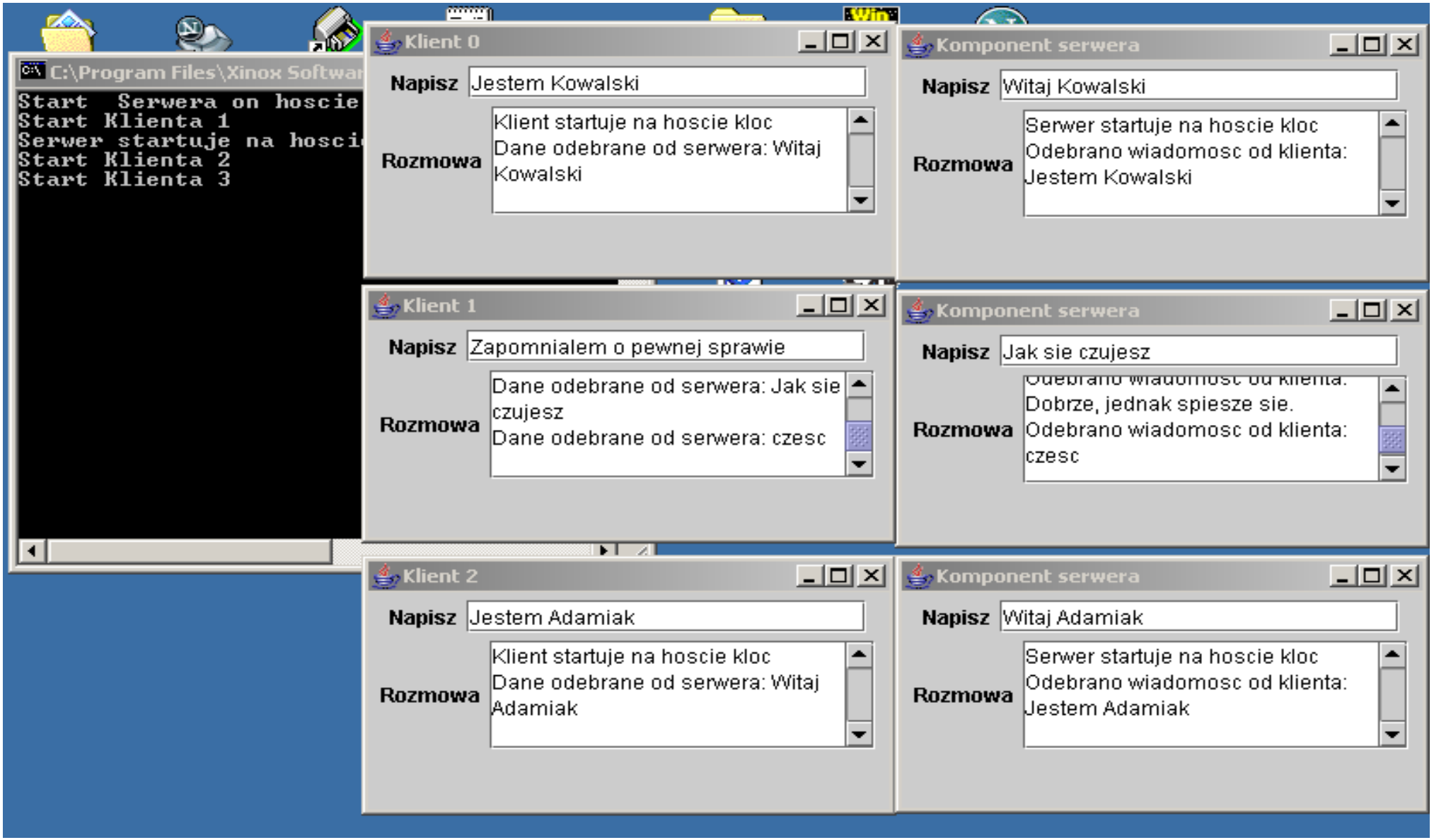

C:\Program Files\Xinox Softwar
Start Serwera on hoscie
Start Klienta 1
Serwer startuje na hosci
Start Klienta 2
Start Klienta 3
Klient 0
Napisz
Jestem Kowalski
Rozmowa
Klient startuje na hoscie kloc
Komponent serwera
Napisz
Rozmowa
Serwer startuje na hoscie kloc
Odebrano wiadomosc od klienta:
Jestem Kowalski
Klient 1
Napisz
Jestem Nowak
Rozmowa
Klient startuje na hoscie kloc
Komponent serwera
Napisz
Rozmowa
Serwer startuje na hoscie kloc
Odebrano wiadomosc od klienta:
Jestem Nowak
Klient 2
Napisz
Jestem Adamiak
Rozmowa
Klient startuje na hoscie kloc
Komponent serwera
Napisz
Rozmowa
Serwer startuje na hoscie kloc
Odebrano wiadomosc od klienta:
Jestem Adamiak


C:\Program Files\Xinox Softwar
Start Serwera on hoscie
Start Klienta 1
Serwer startuje na hosci
Start Klienta 2
Start Klienta 3
Klient 0
Napisz
Jestem Kowalski
Rozmowa
Klient startuje na hoscie kloc
Dane odebrane od serwera: Witaj Kowalski
Komponent serwera
Napisz
Witaj Kowalski
Rozmowa
Serwer startuje na hoscie kloc
Odebrano wiadomosc od klienta: Jestem Kowalski
Klient 1
Napisz
Jestem Nowak
Rozmowa
Klient startuje na hoscie kloc
Dane odebrane od serwera: Witaj Nowak
Komponent serwera
Napisz
Witaj Nowak
Rozmowa
Serwer startuje na hoscie kloc
Odebrano wiadomosc od klienta: Jestem Nowak
Klient 2
Napisz
Jestem Adamiak
Rozmowa
Klient startuje na hoscie kloc
Komponent serwera
Napisz
Witaj Adamiak
Rozmowa
Serwer startuje na hoscie kloc
Odebrano wiadomosc od klienta: Jestem Adamiak


C:\Program Files\Xinox Softwar
Start Serwera on hoscie
Start Klienta 1
Serwer startuje na hosci
Start Klienta 2
Start Klienta 3
Klient 0
Napisz
Jestem Kowalski
Rozmowa
Klient startuje na hoscie kloc
Dane odebrane od serwera: Witaj Kowalski
Komponent serwera
Napisz
Witaj Kowalski
Rozmowa
Serwer startuje na hoscie kloc
Odebrano wiadomosc od klienta: Jestem Kowalski
Klient 1
Napisz
Jestem Nowak
Rozmowa
Dane odebrane od serwera: Witaj Nowak
Dane odebrane od serwera: Jak sie czujesz
Komponent serwera
Napisz
Jak sie czujesz
Rozmowa
Serwer startuje na hoscie kloc
Odebrano wiadomosc od klienta: Jestem Nowak
Klient 2
Napisz
Jestem Adamiak
Rozmowa
Klient startuje na hoscie kloc
Dane odebrane od serwera: Witaj Adamiak
Komponent serwera
Napisz
Witaj Adamiak
Rozmowa
Serwer startuje na hoscie kloc
Odebrano wiadomosc od klienta: Jestem Adamiak


C:\Program Files\Xinox Softwar
Start Serwera on hoscie
Start Klienta 1
Serwer startuje na hosci
Start Klienta 2
Start Klienta 3
Klient 0
Napisz
Jestem Kowalski
Rozmowa
Klient startuje na hoscie kloc
Dane odebrane od serwera: Witaj Kowalski
Komponent serwera
Napisz
Witaj Kowalski
Rozmowa
Serwer startuje na hoscie kloc
Odebrano wiadomosc od klienta: Jestem Kowalski
Klient 1
Napisz
Dobrze, jednak spiesze sie.
Rozmowa
Dane odebrane od serwera: Witaj Nowak
Dane odebrane od serwera: Jak sie czujesz
Komponent serwera
Napisz
Jak sie czujesz
Rozmowa
Odebrano wiadomosc od klienta: Jestem Nowak
Odebrano wiadomosc od klienta: Dobrze, jednak spiesze sie.
Klient 2
Napisz
Jestem Adamiak
Rozmowa
Klient startuje na hoscie kloc
Dane odebrane od serwera: Witaj Adamiak
Komponent serwera
Napisz
Witaj Adamiak
Rozmowa
Serwer startuje na hoscie kloc
Odebrano wiadomosc od klienta: Jestem Adamiak


C:\Program Files\Xinox Softwa
Start Serwera on hoscie
Start Klienta 1
Serwer startuje na hosci
Start Klienta 2
Start Klienta 3
Klient 0
Napisz
Jestem Kowalski
Rozmowa
Klient startuje na hoscie kloc
Dane odebrane od serwera: Witaj Kowalski
Komponent serwera
Napisz
Witaj Kowalski
Rozmowa
Serwer startuje na hoscie kloc
Odebrano wiadomosc od klienta: Jestem Kowalski
Klient 1
Napisz
czesc
Rozmowa
Dane odebrane od serwera: Witaj Nowak
Dane odebrane od serwera: Jak sie czujesz
Komponent serwera
Napisz
Jak sie czujesz
Rozmowa
Dobrze, jednak spiesze sie.
Odebrano wiadomosc od klienta: czesc
Klient 2
Napisz
Jestem Adamiak
Rozmowa
Klient startuje na hoscie kloc
Dane odebrane od serwera: Witaj Adamiak
Komponent serwera
Napisz
Witaj Adamiak
Rozmowa
Serwer startuje na hoscie kloc
Odebrano wiadomosc od klienta: Jestem Adamiak



Koniec komunikacji między klientem 1 a komponentem serwera po otrzymaniu słowa cześć komponent serwera potwierdza odebranie „czesc” wysłaniem słowa „czesc” i kończy połączenie. Komunikat „Zapomnialem o pewnej sprawie” nie zostanie wysłany przez klienta.

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

class server_komp extends JFrame implements Runnable, ActionListener
{  private Socket s;
   private ObjectOutputStream output;
   private ObjectInputStream input;
   private String m = "", m1 = "";
   JTextField nazwa = new JTextField(20);
   JTextArea komentarz = new JTextArea (4,18);

public void actionPerformed( ActionEvent evt)
   { Object zrodlo = evt.getSource();
      if (zrodlo == nazwa)
       { m1 = nazwa.getText();
         if (!m1.equals("czesc")&& s != null)
          try
           {  output.writeObject((Object) m1);
           }
           catch(Exception e)
            {  System.out.println("Wyjatek serwera2 "+e);
            }
       }
      repaint();
   }
```

```
public server_komp(Socket s_,ObjectInputStream input_, ObjectOutputStream output_)
    { super("Komponent serwera");
      s=s_;
      input=input_;
      output=output_;

      setSize(300,160);
      nazwa.addActionListener(this);
      setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
     JPanel panel=new JPanel();

      JLabel etykieta_nazwy = new JLabel ("Napisz");
      JLabel etykieta_komentarza = new JLabel  ("Rozmowa");
      komentarz.setLineWrap(true);
      komentarz.setWrapStyleWord(true);
      panel.add(etykieta_nazwy);
      panel.add(nazwa);
      panel.add(etykieta_komentarza);
      panel.add(komentarz);
      JScrollPane obszar_przewijany1 = new JScrollPane
        ( komentarz,
          ScrollPaneConstants.VERTICAL_SCROLLBAR_ALWAYS,
          ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED);
       panel.add(obszar_przewijany1);
       setContentPane(panel);
       setVisible(true);
    }
```

```
public void run()          //metoda obiektu wykonywana wątku
  {
    String pom;
    try
    {  komentarz.setText("Serwer startuje na hoscie "+
                                        InetAddress.getLocalHost().getHostName()+"\n");
    while(true)
     {
       m = (String) input.readObject();
       pom=komentarz.getText();
       komentarz.setText(pom+"Odebrano wiadomosc od klienta: "+ m +"\n");
       if (m.equals("czesc"))
         {
           m1="czesc";
           output.writeObject((Object) m1);
            break;
         }
      }
      input.close();
      output.close();
      s.close();
      s=null;
    } catch (Exception e)
       {  System.out.println("Wyjatek serwera1 "+e);   }
  }
}
```

```
public class serwer3 implements Runnable
{
public void run()                          //metoda serwera wykonywana wątku - rozpoznaje połączenia
 {   Socket s;                             //z kolejnymi klientami
     ObjectOutputStream output;
     ObjectInputStream input;
     System.out.println("Serwer startuje na hoscie "+host);
     while (true)
      { try
        { s = serwer.accept();             //rozpoznawanie połączenia z klientem
         } catch (IOException e)
            {  System.out.println("Nie mozna polaczyc sie z klientem "+e);
               System.exit(1);             //to  rozwiązanie obsługi wyjątku nie jest zalecane w praktyce!
            }
        if ( s != null)                    //tworzenie strumieni wejścia/wyjścia
         { try                             //oraz wątku z obiektem do obsługi połączenia z kolejnym klientem
           {
              output = new ObjectOutputStream(s.getOutputStream());
              output.flush();
              input = new ObjectInputStream(s.getInputStream());
              Thread t=new Thread(new server_komp (s, input, output));
              t.start();
           } catch (Exception e)
             {  System.out.println("Wyjatek serwera "+e); }
         }
      }
  }
```

```
  private int  sPort;
  private ServerSocket serwer;
  private String host;

 public serwer3(int port_, String host_)
  {
    sPort = port_;
     host=host_;
     try
        {
           serwer = new ServerSocket(sPort);     //serwer tworzy gniazdo do wykrywania
         } catch(IOException e)                  //połączeń z klientami
              {
                 System.out.println(e);  }
    }

//ten program należy uruchomić jako pierwszy

 public static void main(String args[]) throws Exception
  {
    String host_ = InetAddress.getLocalHost().getHostName();
     int Port = 5000;
     serwer3 s2 = new serwer3(Port, host_);
     Thread t = new Thread(s2);
     t.start();
  }
 }
```

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

public class klient3 extends JFrame implements Runnable,  ActionListener
{ private int port;
   private Socket s;
   private ObjectOutputStream output;
   private ObjectInputStream input;
   private String host, m = "", m1 = "";

   JTextField nazwa = new JTextField(20);
   JTextArea komentarz = new JTextArea (4,18);


 public void actionPerformed(ActionEvent evt)
   {  Object zrodlo = evt.getSource();
       if (zrodlo == nazwa)
        { m1 = nazwa.getText();
          if (s != null)
           try
              { output.writeObject((Object) m1);  }
          catch(Exception e)
            {System.out.println("Wyjatek klienta3 "+e);}
        }
      repaint();
   }
```

```
klient3(String host_, int port_, int i)
 {  super("Klient "+i);
     host = host_;
     port = port_;
     setSize(300,160);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    nazwa.addActionListener(this);
    JPanel panel=new JPanel();
    JLabel etykieta_nazwy = new JLabel ("Napisz");
    JLabel etykieta_komentarza = new JLabel  ("Rozmowa");
     komentarz.setLineWrap(true);
     komentarz.setWrapStyleWord(true);
     panel.add(etykieta_nazwy);
     panel.add(nazwa);
     panel.add(etykieta_komentarza);
     panel.add(komentarz);
     JScrollPane obszar_przewijany1 = new JScrollPane
      (komentarz,
        ScrollPaneConstants.VERTICAL_SCROLLBAR_ALWAYS,
        ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED);
    panel.add(obszar_przewijany1);
    setContentPane(panel);
    setVisible(true);
 }
```

```
public void run()       //metoda obiektu wykonywana w wątku
  { String pom;
    try
     { s = new Socket (host, port);
       input = new ObjectInputStream(s.getInputStream());
       output = new ObjectOutputStream(s.getOutputStream());
       output.flush();
       komentarz.setText("Klient startuje na hoscie "+
            InetAddress.getLocalHost().getHostName()+"\n");
     } catch (Exception e)
         {System.out.println("Wyjatek klienta1 "+e);}
    try
     { do
       {    m = (String) input.readObject();
            pom=komentarz.getText();
            komentarz.setText(pom + "Dane odebrane od serwera: " + m + "\n"); }
        } while(!m.equals("czesc"));
      s.close();
      s = null;
      output.close();
      input.close();
     } catch (Exception e)
        {System.out.println("Wyjatek klienta2 "+e);}
}
```

```
//ten program należy uruchomić jako drugi

public static void main(String args[]) throws Exception
    {
       String s = InetAddress.getLocalHost().getHostName();
        klient3 k2 = new klient3(s,5000,1);
       Thread t = new Thread(k2);
       t.start();
    }
}
```

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

class Tester
 {
    public Tester()
     {
        super();
     }
```

/ Program Testera startuje najpierw tworząc obiekt serwera **server** i wstawia go do wątku i następnie tworzy tablicę **clients** zawierającą trzy wątki, każdy z klientem. Po starcie serwer tworzy gniazdo **ServerSocket** o nazwie **serwer** i jego metodą **accept** oczekuje na zgłoszenie klienta. Każdy z klientów po wystartowaniu w niezależnym wątku tworzy gniazdo typu **Socket** znając port i nazwę hosta, na którym znajduje się serwer oraz tworzy strumienie wejścia/wyjścia typu **ObjectOutputStream** o nazwie **output** oraz typu **ObjectInputStream** o nazwie **input** i wysyła do serwera komunikat (np. Jestem Kowalski). Kiedy metoda **accept** wykryje połączenie z klientem, zwraca powiązany z klientem obiekt typu **Socket** o nazwie **s**. Serwer tworzy strumienie wejścia/wyjścia typu **ObjectOutputStream** o nazwie **output** oraz typu **ObjectInputStream** o nazwie **input**. Następnie tworzy obiekt typu **server_komp** i wstawia go wątku pochodnego przekazując mu gniazdo **s** oraz strumienie **input** i **output**. Za jego pośrednictwem może serwer porozumiewać się z klientem i działać jednocześnie niezależnie tzn. identyfikować za pomocą metody **accept** gniazda **ServerSocket** połączenia z nowymi klientami, tworząc nowe wątki pochodne z obiektami typu **server_komp**. /*

```
public static void main(String args[])
    {
        int NUMCLIENTS = 3;
        int sPort=5000;
        String host;
        Thread server;
        Thread clients[]= new Thread[NUMCLIENTS];
        try
          {
            host = InetAddress.getLocalHost().getHostName();
            System.out.println("Start  Serwera on hoscie "+ host);
            server = new Thread(new serwer3(sPort,"Server"), host);
            server.start();
            for (int i=0;i<clients.length;i++)
              {
                System.out.println("Start Klienta " + (i +1));
                clients[i]=new Thread(new  klient3(host,sPort,i));
                clients[i].start();
               }
        } catch (UnknownHostException e)
          {
            System.out.println("Nieznany wyjątek podczas startu klienta");
          }
    }
  }
```