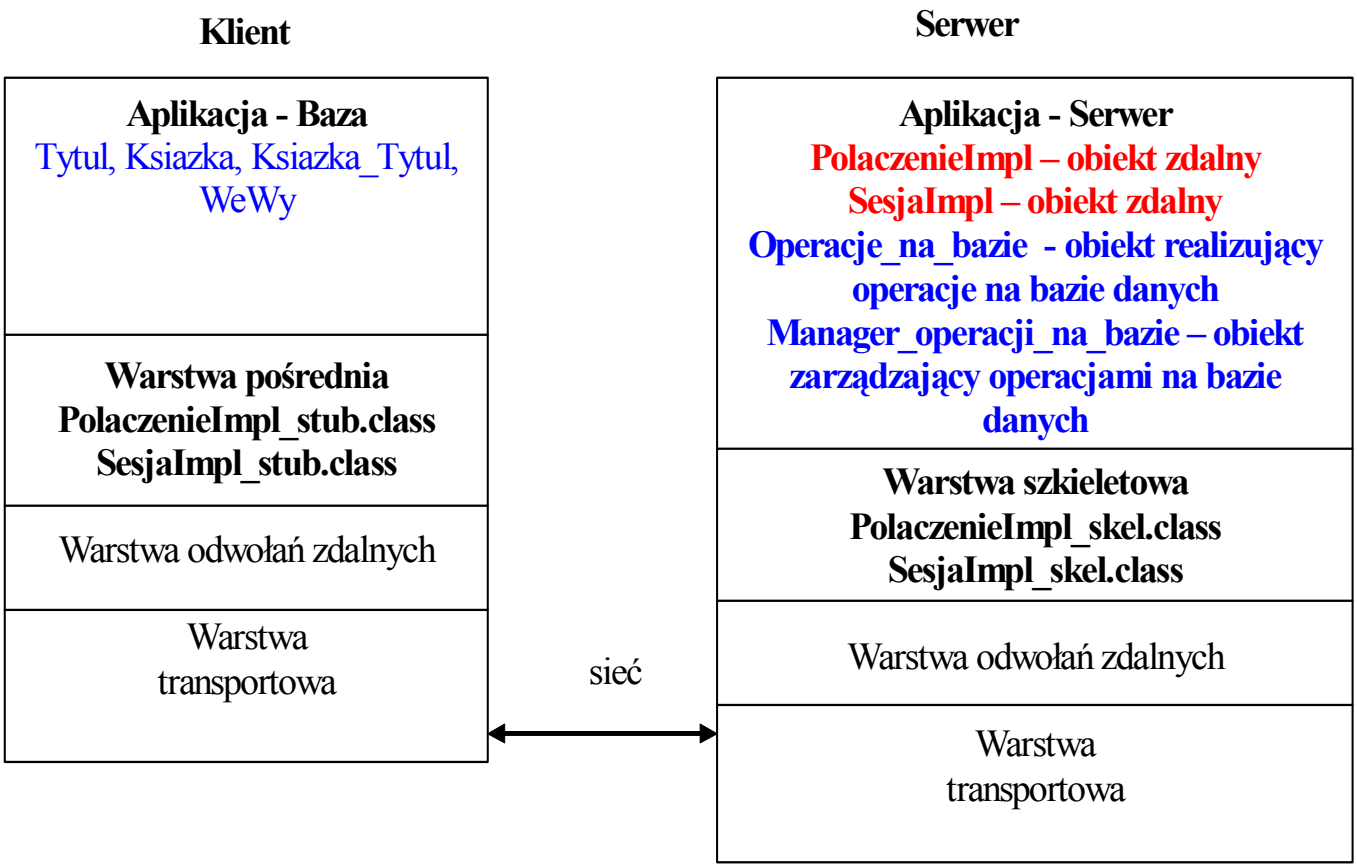
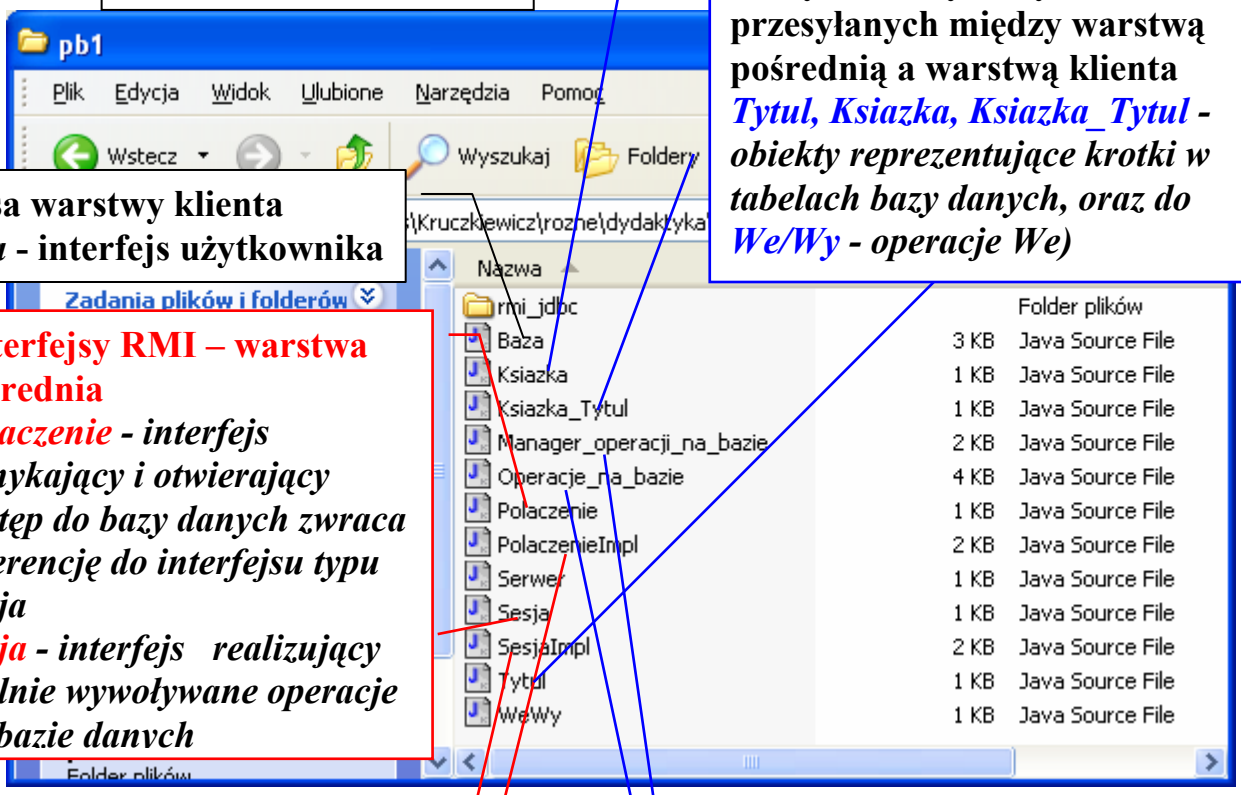
Mechanizmy RMI i JDBC w dostępie do baz danych
(Michał Grochala: „Java – aplikacje bazodanowe”, „Helion”, 2001)



Pliki źródłowe aplikacji



Klasy warstwy danych przesyłanych między warstwą pośrednią a warstwą klienta
Tytul, Ksiazka, Ksiazka_Tytul* - *obiekty reprezentujące krotki w tabelach bazy danych, oraz do
We/Wy* - *operacje We)

Klasa warstwy klienta
***Baza* - interfejs użytkownika**

Interfejsy RMI – warstwa pośrednia
Polaczenie* - *interfejs zamykający i otwierający dostęp do bazy danych zwraca referencję do interfejsu typu Sesja
Sesja* - *interfejs realizujący zdalnie wywoływane operacje na bazie danych

Obiekty zdalne RMI –
obiekty implementujące interfejsy RMI
PolaczenieImpl
SesjaImpl

Klasy warstwy pośredniej
Manager_operacji_na_bazie*-*klasa odpowiedzialna za otwieranie i zamykanie połączeń z bazą danych; udostępnia referencje do obiektu typu Operacje_na_bazie
***Operacje_na_bazie* –**
klasa odpowiedzialna za zapis do bazy i odczyt z bazy danych potrzebnych do tworzenia obiektów z warstwy obiektów; zawiera wywołanie preinterpretowanych zapytań do bazy danych

Program klienta obiektów RMI

```
package przyklad.klient;
import java.io.*;
import java.util.*;
import java.rmi.*;
import przyklad.obiekty.*;
import przyklad.*;

public class Baza
{  String data, sql;
    Polaczenie polaczenie = null;
    Sesja sesja = null;
    String uzytkownik=""; //można wprowadzać login i hasło do programu; w przykładzie
    String haslo=""; // przyjęto dane puste

  void drukuj_tytuly(Tytul t[])
  { int ile=t.length;
     for (int i=0;i<ile;i++)      t[i].drukuj(); }

  void drukuj_ksiazki(Ksiazka_Tytul t[])
  { int ile=t.length;
    for (int i=0;i<ile;i++)      t[i].drukuj();  }

 Tytul podaj_tytul()throws Exception
  { Tytul t = new Tytul();
    t.wstaw_tytul();
    return t;  }

Ksiazka podaj_ksiazke()
   { Ksiazka k = new Ksiazka();
      k.wstaw_ksiazke();
      return k; }


void operacje_na_bazie() throws Exception
  { int opcja;
    do
    { System.out.println("\n"+"1  -  wyswietl tytuly");
      System.out.println("2  -  wyswietl ksiazki");
      System.out.println("3  -  wyszukaj ksiazki danego autora");
      System.out.println("4  -  wstaw tytul");
      System.out.println("5  -  wstaw ksiazke");
      System.out.println("-1 -  koniec programu");
      opcja = WeWy.weInteger("Podaj opcje: ");
```

```
switch(opcja)
    { case 1 :   System.out.println("Tytuly");
                 Tytul t[]=sesja.wyswietl_tytuly();
                 drukuj_tytuly(t);                                  break;
      case 2 :   System.out.println("Ksiazki");
                  Ksiazka_Tytul kt[]=sesja.wyswietl_ksiazki();
                 drukuj_ksiazki(kt);                                break;
      case 3 :   String autor =  WeWy.weString("Podaj autora: ");
                  Ksiazka_Tytul kt_[]= sesja.wyszukaj(autor);
                  drukuj_ksiazki(kt_);                              break;
      case 4 :    sesja.dodaj_tytuly(podaj_tytul());               break;
      case 5 :    sesja.dodaj_ksiazki(podaj_ksiazke());            break;
      case -1:   System.out.println("Koniec programu");           break;
      default:   System.out.println("Zla opcja");
    }
   }while(opcja!=-1);
 }

 void glowna()
 { System.setSecurityManager(new RMISecurityManager());
      try
      { polaczenie =  (Polaczenie)Naming.lookup("Przyklad_polaczenie");
        if  (polaczenie!=null )
          { System.out.println("Jest polaczenie");
            sesja= polaczenie.polacz(uzytkownik, haslo);  //polaczenie, sesja – obiekty zdalne
           if ( sesja!=null)
             {  System.out.println("Jest sesja");
                operacje_na_bazie(); }
           }
       } catch(Exception e)
          { System.out.println("Blad bazy "+e);
            polaczenie=null;
            sesja=null;}
      try
        {  polaczenie.rozlacz(uzytkownik); //koniec połączenia z baza danych
        } catch(Exception e)
            { System.out.println(e.getMessage()); }
      finally
        { polaczenie =null;
          sesja=null;  }
  }
   static  public void main(String arg[])
  { Baza baza = new Baza();
    baza.glowna(); }
}
```

Obiekty pośredniczące między warstwa klienta i warstwą obiektów zdalnych

```
package przyklad.obiekty;
import java.io.*;
import java.util.*;
import java.rmi.*;
import przyklad.*;

public class Tytul implements Serializable
 { public String tytul,autor;
   public int ISBN;
   public Tytul(String _tytul, String _autor, int _ISBN)
   { tytul=_tytul;
     autor=_autor;
     ISBN=_ISBN;}

public Tytul( )  { }

  public void wstaw_tytul() throws Exception
   { tytul = WeWy.weString("Podaj tytul: ");
     autor = WeWy.weString("Podaj autora: ");
     ISBN = WeWy.weInteger("Podaj ISBN: "); }

  public void drukuj()
   { System.out.println(tytul+'\t'+autor+'\t'+ISBN+'\t'); }
 }
**********************************************************************
 public class Ksiazka implements Serializable
{ public int numer;
  public String tytul;
  public Ksiazka(int _numer)
   {numer=_numer;}

  public Ksiazka( )
   { }

  public void wstaw_ksiazke()
  { numer = WeWy.weInteger("Podaj numer ksiazki: ");
    tytul = WeWy.weString("Podaj tytul ksiazki: ");
  }

  public void drukuj()
   {System.out.println(numer+'\t'+tytul+'\t');
   }
}
```

```
package przyklad.obiekty;
import java.io.*;
import java.util.*;
import java.rmi.*;
import przyklad.*;

public class Ksiazka_Tytul extends Tytul        implements Serializable
 { public int  numer;
   public Ksiazka_Tytul(String _tytul, String _autor, int _ISBN, int _numer)
   { super(_tytul,_autor,_ISBN);
     numer=_numer; }

   public void drukuj()
   { System.out.print(numer+"   "+'\t');
      super.drukuj(); }
}
**********************************************************************
package przyklad.obiekty;
import java.io.*;
import java.util.*;
import java.rmi.*;
import przyklad.*;

public class WeWy
{ public static String weString(String menu)
    { InputStreamReader wejscie = new InputStreamReader( System.in );
      BufferedReader bufor = new BufferedReader( wejscie );
     try
       { System.out.print(menu);
         return bufor.readLine();
       }catch (IOException e)
         { System.err.println("Blad IO String");
           return ""; } }

   public static byte weInteger(String menu)
    { InputStreamReader wejscie = new InputStreamReader( System.in );
      BufferedReader bufor = new BufferedReader( wejscie );
      StringTokenizer zeton;
      try
        { System.out.print(menu);
          zeton = new StringTokenizer(bufor.readLine());
          return Byte.parseByte(zeton.nextToken());
        } catch (Exception e)
           { System.err.println("Blad Integer "+e);
             return 0;  } } }
```

Interfejsy i obiekty RMI

```
package przyklad;
import java.rmi.*;
import przyklad.obiekty.*;

public interface Sesja extends Remote  //interfejs RMI
{
  public Tytul[] wyswietl_tytuly()              throws RemoteException;
  public Ksiazka_Tytul[] wyswietl_ksiazki()     throws RemoteException;
  public void dodaj_tytuly(Tytul t)             throws RemoteException;
  public void dodaj_ksiazki(Ksiazka k)          throws RemoteException;
  public Ksiazka_Tytul[] wyszukaj(String a)     throws RemoteException;
}
```

****

```
package przyklad;
import java.rmi.*;
import java.rmi.server.*;
import przyklad.serwer.operacje.*;
import przyklad.klient.*;
import przyklad.obiekty.*;


class SesjaImpl extends UnicastRemoteObject  implements Sesja
 {
   protected Manager_operacji_na_bazie manager_operacji_na_bazie = null;
     //obiekt pomocniczy zarządzający operacjami na bazie danych
   public SesjaImpl(String drivers, String dbUrl, String uzytkownik, String haslo)
                                                          throws RemoteException
    {manager_operacji_na_bazie =
    new Manager_operacji_na_bazie(drivers, dbUrl, uzytkownik, haslo);}

   public boolean otworz_baze_danych (String uzytkownik, String haslo)
    {return manager_operacji_na_bazie.otwarcie_polaczenia(uzytkownik, haslo);}

   public void zamknij_baze_danych()
    {manager_operacji_na_bazie.zamkniecie_polaczenia();}

   public Tytul[] wyswietl_tytuly() throws RemoteException
    { try
      {return
         manager_operacji_na_bazie.podaj_Operacje_na_bazie().wyswietl_tytuly();
       }catch(Exception e)
       {}
      return null; }
```

```
public  Ksiazka_Tytul[] wyswietl_ksiazki() throws RemoteException
    {try
      {return
        manager_operacji_na_bazie.podaj_Operacje_na_bazie().wyswietl_ksiazki();
      }catch(Exception e)
       {}
     return null;
    }

  public void dodaj_tytuly(Tytul t) throws RemoteException
    {try
      {  manager_operacji_na_bazie.podaj_Operacje_na_bazie().wstaw_tytul(t);
       }catch(Exception e)
          {}
    }

  public void dodaj_ksiazki(Ksiazka k) throws RemoteException
    {try
     {manager_operacji_na_bazie.podaj_Operacje_na_bazie().wstaw_ksiazke(k);
      }catch(Exception e)
         {}
    }

  public  Ksiazka_Tytul[] wyszukaj(String a) throws RemoteException
    {try
      {return manager_operacji_na_bazie.podaj_Operacje_na_bazie().wyszukaj(a);
      }catch(Exception e)
        {}
     return null;
    }
  }
**********************************************************************
package przyklad;
import java.rmi.*;
 public interface Polaczenie extends Remote
 { public Sesja polacz(String uzytkownik, String haslo)  throws RemoteException;
   public void rozlacz (String uzytkownik) throws RemoteException;
 }
**********************************************************************
package przyklad;
import java.rmi.*;
import java.rmi.server.*;
import java.util.*;
import java.net.*;
import przyklad.serwer.operacje.*;
```

```
public class PolaczenieImpl extends UnicastRemoteObject
                                                    implements Polaczenie
{
      Properties properties = null;
      final String dbUrl = "jdbc:odbc:katalog";
      final String drivers = "sun.jdbc.odbc.JdbcOdbcDriver";

      public PolaczenieImpl() throws RemoteException
       {properties = new Properties(); }

      public synchronized Sesja polacz (String uzytkownik, String haslo)
                                              throws RemoteException
       { if (properties.containsKey(uzytkownik))
          {   System.out.println("user");
              return (SesjaImpl) properties.get(uzytkownik); }
         else
          { SesjaImpl sesja=new SesjaImpl(drivers, dbUrl, uzytkownik, haslo);
            if (sesja.otworz_baze_danych(uzytkownik,haslo))
             {try
              { Naming.rebind("Przyklad"+uzytkownik,sesja);
                properties.put(uzytkownik,sesja);
                System.out.println("Po otwarciu bazy danych");
                return sesja;
              }catch(MalformedURLException e)
                 { System.out.println(e.getMessage());
                   return null; }
            }
             else
              { return null; }
          }
         }

       public synchronized void rozlacz (String uzytkownik)
                                                    throws RemoteException
        { SesjaImpl sesja=(SesjaImpl)properties.get(uzytkownik);
          if (null != sesja)
           { sesja.zamknij_baze_danych ();
             properties.remove(uzytkownik);
             try
              { Naming.unbind("Przyklad"+uzytkownik);
               } catch(Exception e)
                  { System.out.println(e);  }
            }
         }
  }
```

Program serwera obiektów RMI

****

```
package przyklad.serwer;
import java.rmi.*;
import przyklad.*;

public class Serwer
 {
     public Serwer()
       {}

 public void prepareInterface()
  { try
    {  PolaczenieImpl polaczenie= new PolaczenieImpl();
       Naming.rebind("Przyklad_polaczenie",polaczenie);
       System.out.println("Interface gotowy...");
     } catch (Exception e)
          {System.out.println(e);}
  }

 public static void main(String[] args)
  { Serwer serwer = new Serwer();
    System.setSecurityManager(new RMISecurityManager());
    serwer.prepareInterface();
  }
}
```

Obiekty pomocnicze operujące na bazie danych, użyte do definicji metod obiektu RMI typu SesjaImpl

****

```
package przyklad.serwer.operacje;
import przyklad.obiekty.*;
import java.sql.*;
import java.util.*;
import java.io.*;

public class Operacje_na_bazie
 { private Connection  connection = null;

   public Operacje_na_bazie(Connection _connection)
    { connection = _connection;}
  public Tytul[] wyswietl_tytuly() throws Exception
    { Statement polecenie=  connection.createStatement();
      String sql="SELECT * FROM Tytul ORDER BY tytul;";
      ResultSet krotka= polecenie.executeQuery(sql);
      Vector <Tytul>vector= new Vector<Tytul> (10,5);
      while(krotka.next())
       { Tytul tytul = new Tytul( krotka.getString("tytul"),  krotka.getString("autor"),
                                  krotka.getInt("ISBN"));
          vector.addElement(tytul); }
      vector.trimToSize();
      Tytul tytuly[]=new Tytul[vector.size()];
      vector.copyInto(tytuly);
      polecenie.close();
      return tytuly;}

  public Ksiazka_Tytul[] wyswietl_ksiazki() throws Exception
   { Statement polecenie =  connection.createStatement();
     String sql="SELECT * FROM Tytul, Ksiazka  WHERE id_tytul=id_tytul_ "
                 +" ORDER BY tytul;";
     ResultSet krotka = polecenie.executeQuery(sql);
     Vector <Ksiazka>vector=new Vector<Ksiazka> (10,5);
     while(krotka.next())
     {  Ksiazka_Tytul ksiazka_tytul=
          new Ksiazka_Tytul (krotka.getString("tytul"), krotka.getString("autor"),
                             krotka.getInt("ISBN"), krotka.getInt("numer"));
          vector.addElement(ksiazka_tytul);}
      vector.trimToSize();
      Ksiazka_Tytul ksiazki_tytuly[]=new Ksiazka_Tytul[vector.size()];
      vector.copyInto(ksiazki_tytuly);
      polecenie.close();
      return ksiazki_tytuly;  }
```

```
public  Ksiazka_Tytul[] wyszukaj(String autor) throws Exception
  { Statement polecenie = connection.createStatement();
    String sql="SELECT * FROM Tytul, Ksiazka  WHERE id_tytul=id_tytul_"+
                  " AND autor = '" + autor +"' ORDER BY tytul;";
    ResultSet krotka = polecenie.executeQuery(sql);
    Vector <Ksiazka_Tytul>vector = new Vector<Ksiazka_Tytul> (10,5);
    while(krotka.next())
       { Ksiazka_Tytul ksiazka_tytul=  new Ksiazka_Tytul(krotka.getString("tytul"),
           krotka.getString("autor"), krotka.getInt("ISBN"), krotka.getInt("numer"));
         vector.addElement(ksiazka_tytul); }
     vector.trimToSize();
     Ksiazka_Tytul ksiazki_tytuly[]=new Ksiazka_Tytul[vector.size()];
     vector.copyInto(ksiazki_tytuly);
     polecenie.close();
     return ksiazki_tytuly;
  }
 public void wstaw_tytul(Tytul t) throws Exception
  { connection.setAutoCommit(false);
    try
    { Statement polecenie = connection.createStatement();
      String sql="INSERT INTO Tytul (tytul, autor, ISBN)"+
                    " VALUES ('"+t.tytul+"','"+ t.autor+"','"+ t.ISBN+"')";
     polecenie.addBatch(sql);
     polecenie.executeBatch();
     connection.commit();
    } catch(BatchUpdateException e)
          { System.out.println("Wycofanie transakcji");
            connection.rollback(); }
  }
public void wstaw_ksiazke(Ksiazka k) throws Exception
  { connection.setAutoCommit(false);
    try
     { Statement polecenie =  connection.createStatement();
       String sql="SELECT * FROM Tytul  WHERE tytul= '"+k.tytul+"';";
       ResultSet krotka=polecenie.executeQuery(sql);
       if (!krotka.next()) return;
       sql="INSERT INTO Ksiazka (numer, id_tytul_)"+
             " VALUES ('"+k.numer+"','"+ krotka.getString("id_tytul")+"');";
      polecenie.addBatch(sql);
      polecenie.executeBatch();
      connection.commit();
     } catch(BatchUpdateException e)
           { System.out.println("Wycofanie transakcji");
             connection.rollback();}  }
}
```

```
package przyklad.serwer.operacje;
import java.sql.*;
import java.util.*;
import java.io.*;

public class Manager_operacji_na_bazie
{ protected Connection connection = null;
  protected String _dbUrl = null;
  protected Operacje_na_bazie operacje_na_bazie = null;

  public Manager_operacji_na_bazie(String drivers, String dbUrl,
                                   String uzytkownik, String haslo)
   {_dbUrl = dbUrl;
     try
      { Class.forName(drivers);
        connection = DriverManager.getConnection(_dbUrl, uzytkownik, haslo);
      } catch(Exception e)    {}
    }

  public boolean otwarcie_polaczenia(String uzytkownik, String haslo)
   {try
     { connection=DriverManager.getConnection(_dbUrl, uzytkownik, haslo);
       connection.setAutoCommit(false);
       operacje_na_bazie = new Operacje_na_bazie (connection);
     } catch(Exception e)
       { connection = null;
         operacje_na_bazie = null;
         return false;  }
     return true;
   }

   public void zamkniecie_polaczenia()
   { if( null != connection)
      { try
        { connection.close();
         } catch (SQLException e)  { }
       finally
        { connection = null;
          operacje_na_bazie = null; } } }

  public  Operacje_na_bazie podaj_Operacje_na_bazie()
   { return operacje_na_bazie; }
}
```

Pliki wsadowe – uruchomienie na lokalnym komputerze

1) Utworzenie warstw pośredniczących RMI (przykład.SesjaImpl oraz przykład.PolaczenieImpl) oraz uruchomienie programu rmiregistry

Lister - [c:\Documents and Settings\Kruczkiewicz\rozne\dydaktyka\pai\JDBC_RM...

Plik Edytuj Opcje Pomoc 100 %

```
set classpath=
"c:\Program Files\Java\jdk1.5.0_11\bin\rmic" przyklad.SesjaImpl

set classpath=
"c:\Program Files\Java\jdk1.5.0_11\bin\rmic"  przyklad.PolaczenieImpl

start set classpath=
"c:\Program Files\Java\jdk1.5.0_11\bin\rmiregistry"

pause
```

2) Uruchomienie serwera wraz\z obiektami zdalnymi

Lister - [c:\Documents and Settings\Kruczkiewicz\rozne\dydaktyka\pai\JDBC_RMI\pb1\rm...

Plik Edytuj Opcje Pomoc 100 %

```
set classpath=
"c:\Program Files\Java\jdk1.5.0_11\bin\java" -Djava.security.policy=policy.txt
przyklad.serwer.Serwer
pause
```

3) Uruchomienie klienta obiektów zdalnych

Lister - [c:\Documents and Settings\Kruczkiewicz\rozne\dydaktyka\pai\JDBC_RMI\pb1\r...

Plik Edytuj Opcje Pomoc 100 %

```
set classpath=
"c:\Program Files\Java\jdk1.5.0_11\bin\java" -Djava.security.policy=policy.txt
przyklad.klient.Baza
pause
```

4) Zawartość pliku z uprawnieniami aplikacji RMI dla serwera i klienta

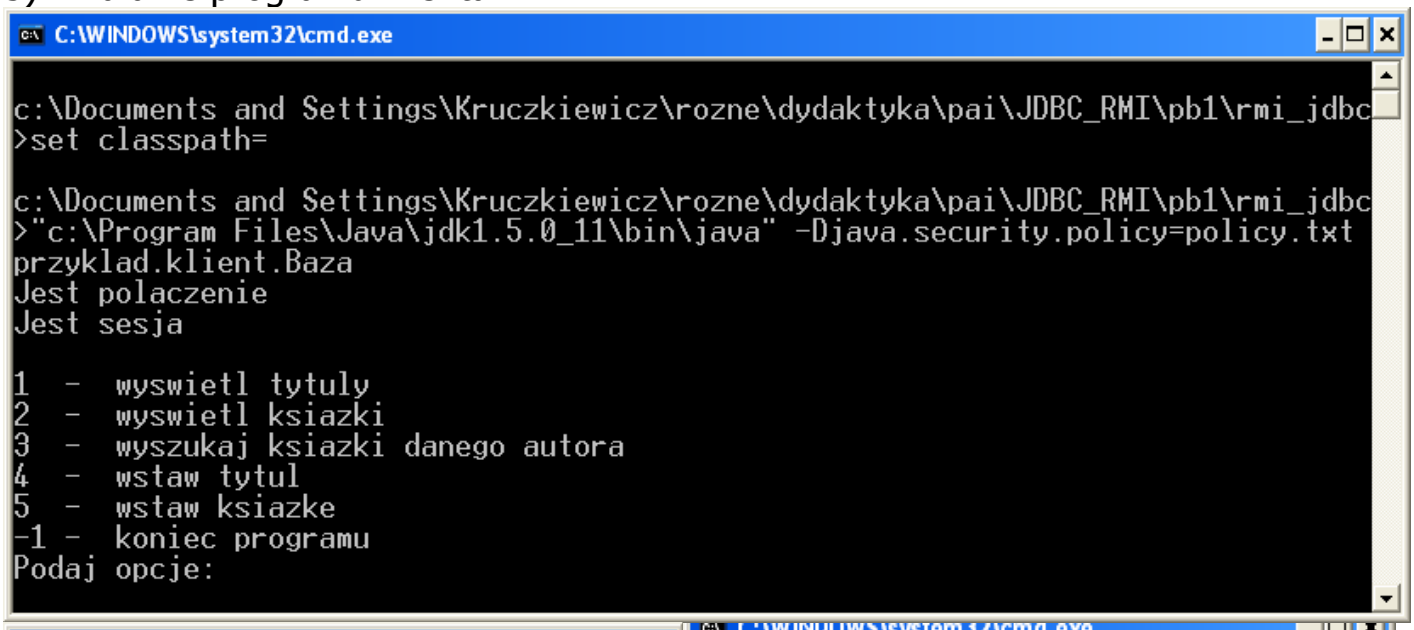
Lister - [c:\Documents and Settings\Kruc...

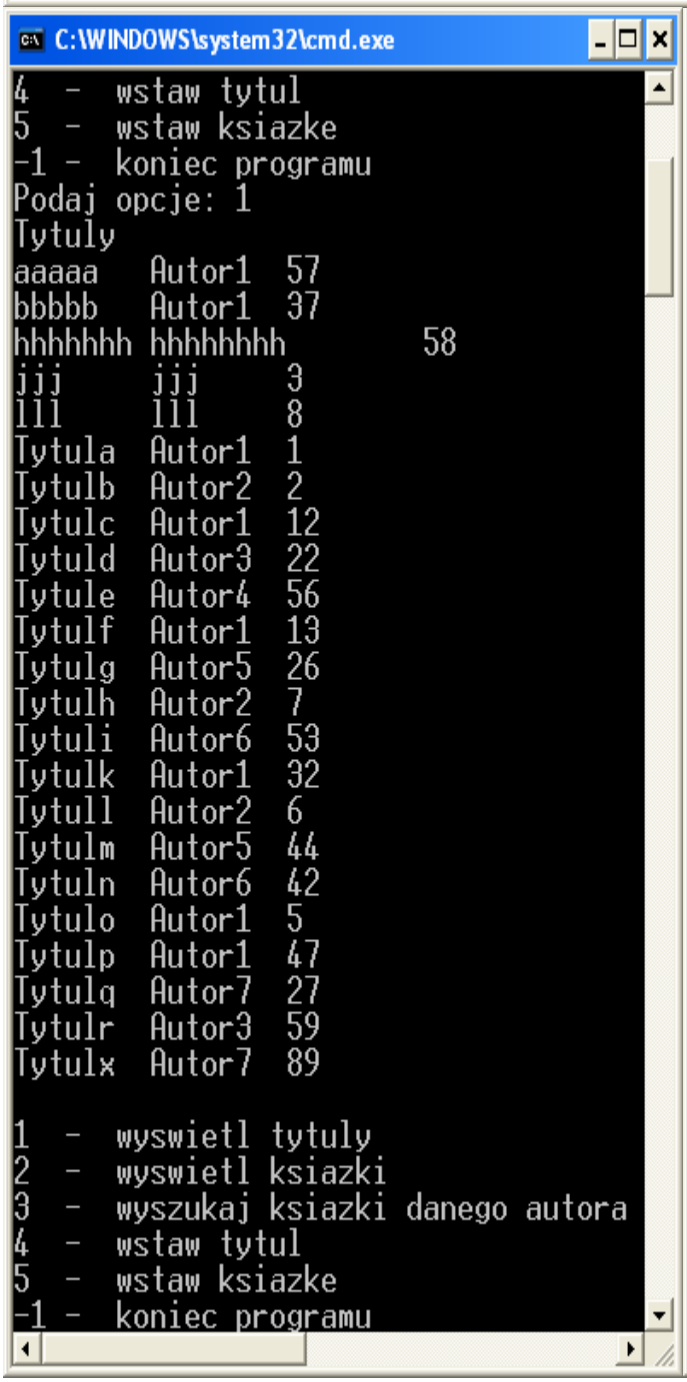
Plik Edytuj Opcje Pomoc 100 %

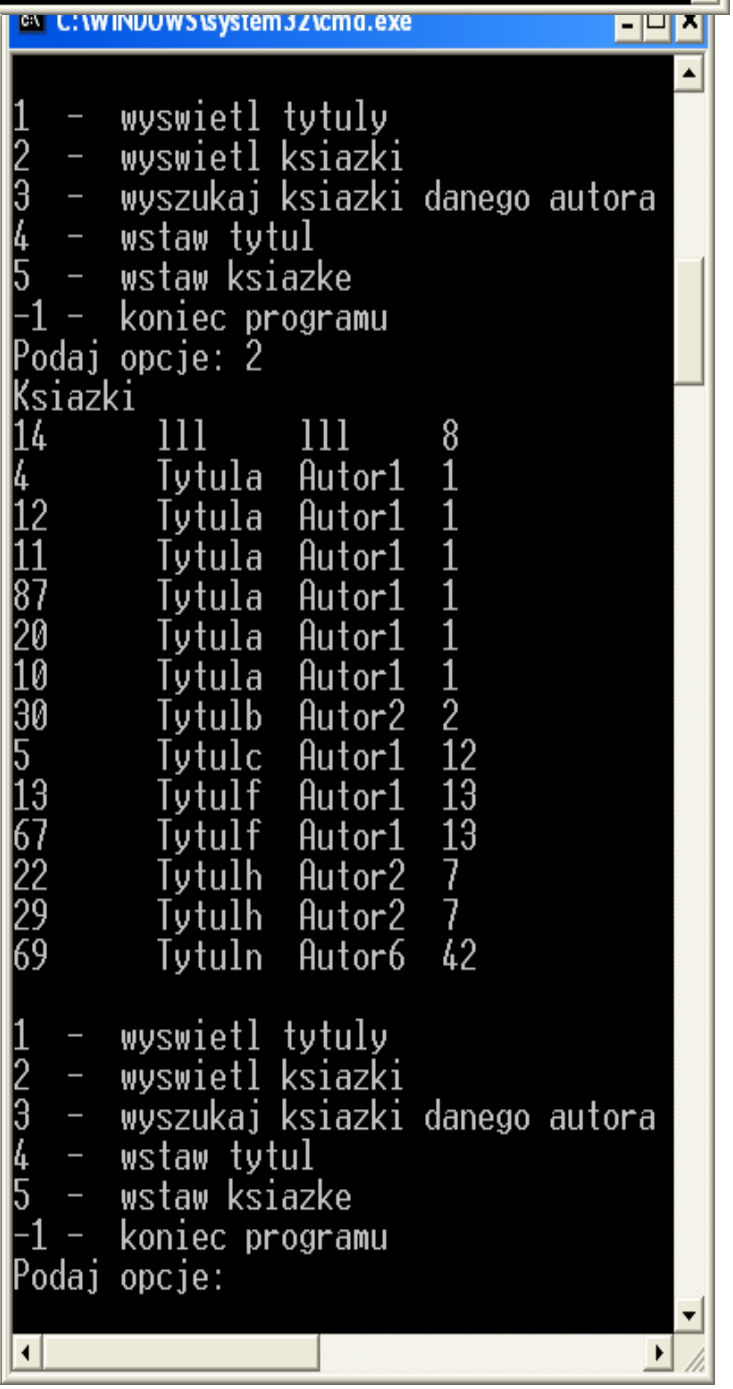
```
grant
{permission java.security.AllPermission;

};
```

5) Działanie programu klienta







6) Działanie programu serwera