



Obezbeđenje pristupa sistemu baze podataka

*Protokoli, ODBC/JDBC, ADO.NET,
OLE DB, O/R prevođenje*

Sadržaj

- Protokoli
- ODBC
- JDBC
- OLE DB
- ADO .NET
- Objektno-relaciono prevođenje
- Hibernate
- Entity Framework

Protokoli

- **Pristup bazi podataka**

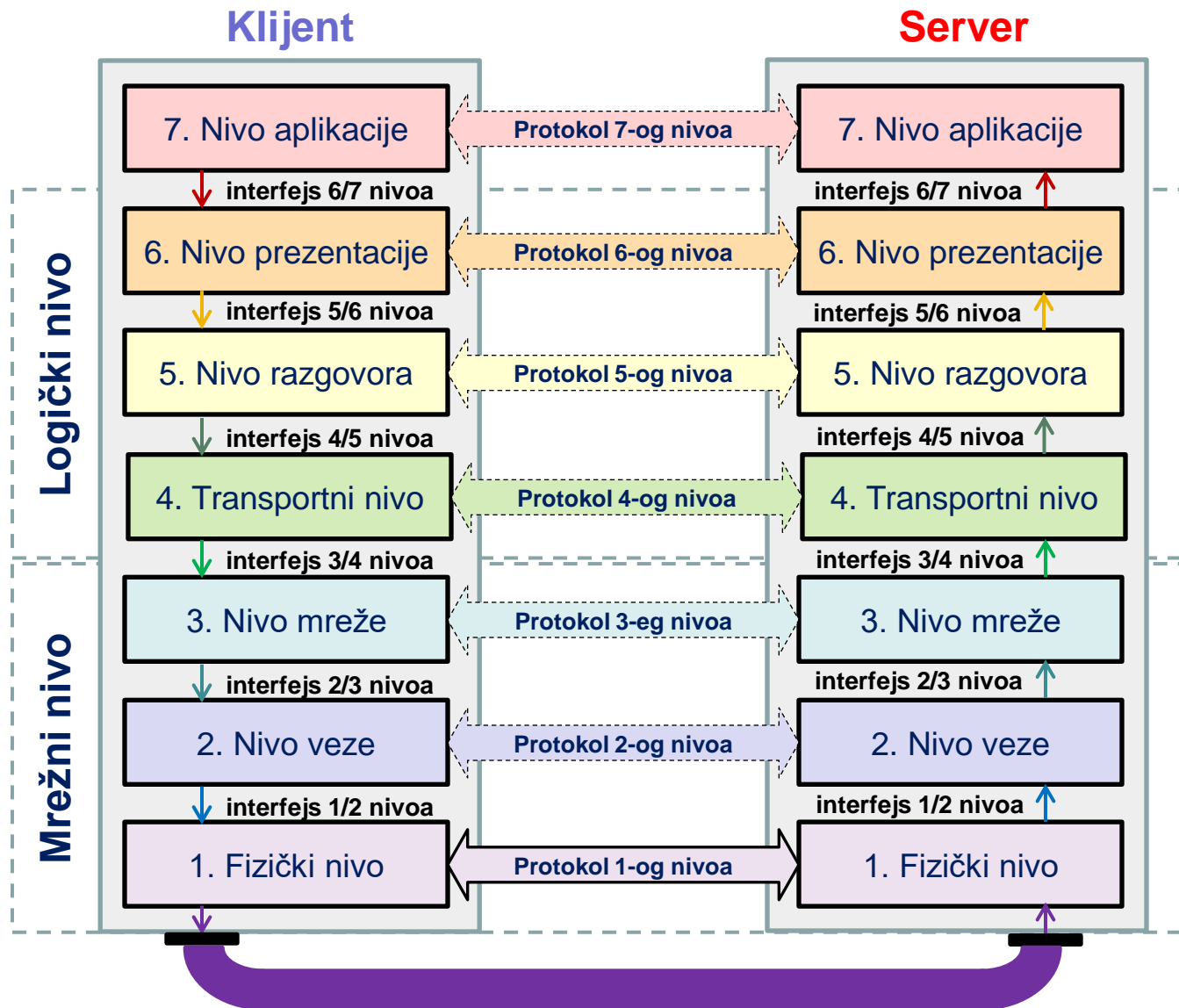
- direktan (eng. *native*) pristup

- pristupa se direktno ciljnom SUBP-u
 - posebna komponenta SUBP-a za prihvatanje podataka od korisničke aplikacije
 - npr. Oracle SQL*Net
 - SQL naredbe se formiraju u sintaksi ciljnog SUBP-a

- indirektan pristup

- uvodi se srednji sloj
 - posrednik u komunikaciji između aplikacije i SUBP-a
 - SQL sintaksa nezavisna od ciljnog SUBP-a
 - direktno korišćenje koncepata relacionog modela podataka u aplikacijama
 - prevođenje objektnih u relacione koncepte

Protokoli



Protokoli

- **ISO/OSI model**

- **1. fizički nivo**

- fizički prenos podataka (0/1 nizova) bez interpretacije
 - putem hardversko-komunikacione infrastrukture

- **2. nivo veze**

- prenos i upravljanje prenosom okvira od čvora do čvora

- **3. nivo mreže**

- funkcionalno obezbeđenje transfera podataka na nivou datoteka OS-a

- **4. nivo**

- prenos i distribuiranje poruka / podataka po sesijama

- **5. nivo**

- uspostava i održavanje komunikacije (sesije)

Protokoli

- **ISO/OSI model**

- **6. nivo prezentacije**

- konceptualno transformisanje poruka i podataka
 - formatiranje, konverzija, kompresija i kodiranje poruka i podataka

- nivoi 4 – 6

- opredeljeni izborom
 - SUBP, na serveru BP
 - » s podrškom jezika SQL i standarda SQL3
 - aplikacionog servera, na srednjem sloju
 - **protokoli**
 - » **JDBC, ODBC**
 - » SOAP, RMI, CORBA/IIOP, HTTP

Protokoli

- **ISO/OSI model**
 - **7. nivo aplikacije**
 - međusobna komunikacija klijentskih i serverskih procesa
 - na klijentu: korisnička aplikacija
 - na serveru: SUBP, koji opslužuje deo ili celu BP IS-a
 - opredeljen izborom razvojnog i izvršnog okruženja aplikacije

Sadržaj

- Protokoli
- ODBC
- JDBC
- OLE DB
- ADO .NET
- Objektno-relaciono prevođenje
- Hibernate
- Entity Framework

ODBC

- **Motivacija**

- kompanije neretko koriste SUBP-ove različitih proizvođača i verzija
- razvoj personalnih računara
 - donosi veliku količinu različitih alata za upite, analizu i prikaz podataka
- razvoj klijent-server arhitekture
 - kompleksnija uspostava i održavanje veze između klijenta i SUBP-a koji se nalaze na različitim računarima

ODBC

- **Motivacija**

- **problem**

- programeri pišu posebne aplikacije za svaki SUBP
 - velika količina resursa troši se na izradu i održavanje aplikacija
 - korisnički zahtevi nameću potrebu da iste aplikacije simultano rade nad različitim SUBP

- **cilj**

- obezbeđenje objedinjenih koncepata i tehnika za pristup različitim SUBP-ovima

ODBC

- **ODBC**

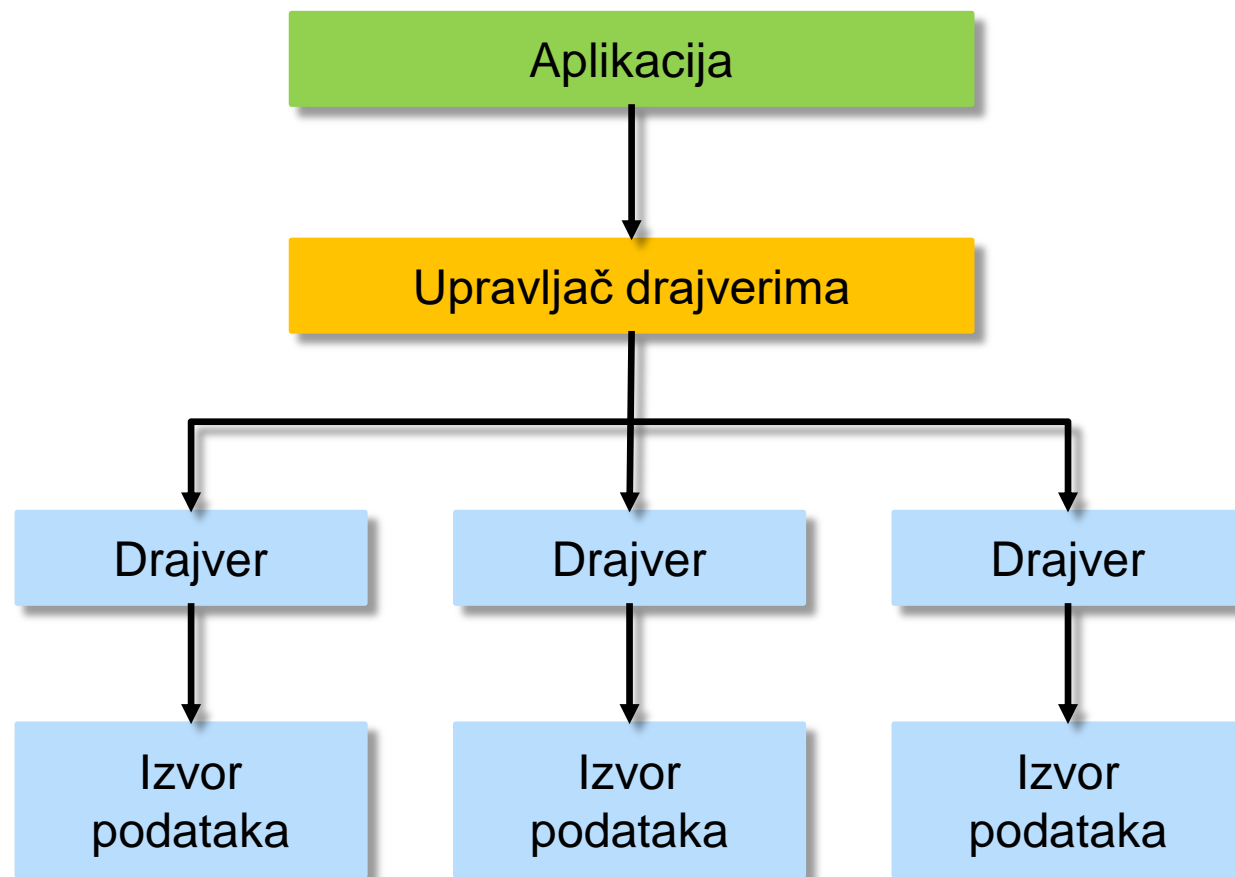
- *eng. Open Database Connectivity*
- inicijalno razvijen od strane Microsofta
 - 1990. godine

- **ODBC API**

- standardni API za pristup bazama podataka
 - skup interfejsa koje proizvođač mora implementirati kako bi napravio drajver za svoju bazu podataka
 - napravljen kako bi pružio
 - objedinjeni pristup različitim SUBP
 - jedinstveno rukovanje greškama
- nezavisan od proizvođača SUBP
- nezavisan od operativnog sistema

ODBC

- **Arhitektura ODBC sistema**



ODBC

- **Arhitektura ODBC sistema**

- **aplikacija**

- poziva ODBC funkcije i procedure
 - zadaci aplikacija
 - odabir i uspostava veze sa izvorom podataka
 - slanje SQL naredbe koja treba da se izvrši
 - preuzimanje rezultata (ukoliko postoje)
 - obrada grešaka
 - commit ili rollback transakcije u kojoj se SQL naredbe nalaze
 - prekid uspostavljene veze sa izvorom podataka

ODBC

- **Arhitektura ODBC sistema**

- **upravljač drajverima**

- *eng. Driver Manager*
 - biblioteka koja upravlja komunikacijom između aplikacije i drajvera
 - olakšava posao programerima prilikom
 - » izbora drajvera
 - » učitavanja drajvera
 - » poziva funkcija drajvera
 - » rada sa višestrukim drajverima
 - prilikom učitavanja svakog drajvera kreira tabelu sa pokazivačima na funkcije datog drajvera
 - putem kojih poziva funkcije drajvera

ODBC

- **Arhitektura ODBC sistema**

- **drajver**

- biblioteka koja implementira funkcije ODBC API-ja
 - svaki drajver je specifičan za određeni SUBP
 - podržava idealno sve funkcije ili, praktično, veći deo funkcija koje poseduje sam SUBP
 - funkcionalnosti drajvera ograničene su mogućnostima ciljnog SUBP-a
 - obrađuje pozive ODBC funkcija
 - prosleđuje SQL naredbe specifičnom izvoru podataka
 - modifikuje SQL naredbu kako bi odgovarala sintaksi specifičnog izvora podataka
 - prosleđuje rezultat aplikaciji

ODBC

- **Arhitektura ODBC sistema**

- **drajver**

- zadaci

- otvaranje i zatvaranje veze sa izvorom podataka
 - provera grešaka u pozivu funkcije
 - » koje nije proverio upravljač drajverima
 - započinjanje transakcije
 - » na zahtev aplikacije
 - prosleđivanje SQL naredbe izvoru podataka
 - » konverzija ODBC SQL naredbi u SQL naredbe specifične za dati SUBP
 - preuzimanje rezultata izvršenja SQL naredbe
 - » konverzija tipova podataka
 - prevođenje SUBP grešaka u ODBC greške
 - » omogućava univerzalnu obradu grešaka

ODBC

- **Arhitektura ODBC sistema**

- **drajver**

- tipovi

- **drajveri za rad sa datotekama**

- » pristupaju fizičkim strukturama podataka neposredno
 - » obuhvataju i servise metode pristupa datotekama
 - » izvršavaju i ODBC funkcije i same SQL naredbe

- **drajveri za rad sa SUBP-ovima**

- » pristupaju strukturama podataka posredno
 - » koristeći servise SUBP
 - » izvršavaju samo ODBC funkcije
 - » SQL naredbe prosleđuju izvoru podataka

ODBC

- **Arhitektura ODBC sistema**

- **izvor podataka**

- obuhvata H/S platformu sa operativnim sistemom, sistem za upravljanje podacima i same podatke
 - fizički organizovane na eksternim memorijskim uređajima
 - ne mora biti na istom računaru kao i drajver
 - može predstavljati
 - sistem baze podataka ili
 - sistem datoteka

ODBC

- **Arhitektura ODBC sistema**
 - dozvoljava istovremenu upotrebu više drajvera i izvora podataka
 - dozvoljava istovremeni pristup različitim izvorima podataka
 - ODBC API se koristi na dva mesta
 - između aplikacije i upravljača drajverima
 - za učitavanje drajvera i uspostavljanje veze
 - između upravljača drajverima i drajvera
 - SPI (*eng. Service Provider Interface*)
 - identičan interfejs za sve drajvere
 - poziv funkcija drajvera

ODBC

- **Primer upotrebe ODBC sistema**
 - primer korišćenja ODBC API-ja na .NET platformi
 - u C# programskom jeziku
 - ODBC .NET Data Provider
 - dodatak .NET razvojnom okruženju
 - sadrži klase za upravljanje komunikacijom sa bazom podataka

ODBC

- **Primer upotrebe ODBC sistema**
 - ODBC .NET Data Provider klase
 - **OdbcConnection**
 - inicijalizuje novu vezu sa ODBC izvorom podataka
 - » koristi se ime izvora podataka
 - » koje mu je dodeljeno prilikom instalacije
 - **OdbcCommand**
 - izvršava naredbu u okviru uspostavljene veze
 - **OdbcDataReader**
 - dozvoljava iteraciju nad rezultatom naredbe
 - izvršava upite nad izvorom podataka

ODBC

- **Primer upotrebe ODBC sistema**
 - ODBC .NET Data Provider klase
 - **OdbcParameter**
 - vezuje parametre za komande
 - **OdbcDataAdapter**
 - popunjava DataSet
 - » reprezentacija podataka u memoriji
 - **OdbcCommandBuilder**
 - kreira Insert, Update i Delete naredbe za OdbcDataAdapter

ODBC

- **Primer upotrebe ODBC sistema**

```
using System;
using Microsoft.Data.Odbc;
namespace BuilderODBC {
    class TestClass {
        static void Main(string[] args) {
            string connectionString = "DSN=TestSourceName;UID=username;Pwd=password;";
            string sql = "SELECT CustomerID, ContactName, ContactTitle FROM Customers";
            OdbcConnection conn = new OdbcConnection(connectionString);
            conn.Open();
            OdbcCommand comm = new OdbcCommand(sql, conn);
            OdbcDataReader dr = comm.ExecuteReader();
            while (dr.Read()) {
                Console.WriteLine(dr.GetValue(0).ToString());
                Console.WriteLine(dr.GetValue(1).ToString());
                Console.WriteLine(dr.GetValue(2).ToString());
            }
            conn.Close();
            dr.Close();
            comm.Dispose();
            conn.Dispose();
        }
    }
}
```

ODBC

- **Primer upotrebe ODBC sistema**
 - rukovanje izuzecima
 - klasa **OdbcException**
 - rukovanje izuzecima koji mogu da se dogode usled interakcije sa ODBC izvorima podataka

ODBC

- **Primer upotrebe ODBC sistema**

```
static void Main(string[] args) {  
    string connectionString = "DSN=Test;UID=Chester;Pwd=Tester;";  
    string sql = "SELECT CustomerID, ContactName, ContactTitle FROM Customers";  
    OdbcConnection conn = null;  
    OdbcCommand comm = null;  
    OdbcDataReader dr = null;  
    try {  
        conn= new OdbcConnection(connectionString);  
        conn.Open();  
        comm = new OdbcCommand(sql, conn);  
        dr = comm.ExecuteReader();  
        while (dr.Read()) { /* čitaj podatke */ }  
        dr.Close();  
    } catch (OdbcException oe) {  
        Console.WriteLine("An ODBC exception occurred: " + oe.Message.ToString());  
    } catch (Exception e) {  
        Console.WriteLine("An exception occurred: " + e.Message.ToString());  
    } finally {  
        if (conn.State == System.Data.ConnectionState.Open)  
            conn.Close();  
        comm.Dispose();  
        conn.Dispose();  
    }  
}
```

Sadržaj

- Protokoli
- ODBC
- JDBC
- OLE DB
- ADO .NET
- Objektno-relaciono prevođenje
- Hibernate
- Entity Framework

JDBC

- **Motivacija**

- sve značajnija uloga Interneta u svakodnevnom životu
- sve značajnija uloga Jave kao programskog jezika
 - nezavisnog od platforme
- lako pravljenje Java aplikacija za rukovanje bazama podataka

JDBC

- **Motivacija**

- **problem**

- nema jedinstvenog načina pristupanja bazi podataka iz Java programa
 - prva verzija JDK-a nije imala podršku za pristup bazama podataka

- **cilj**

- kreirati jedinstven skup funkcija (API) za pristup bilo kojoj bazi podataka
 - iz Java programa

JDBC

- **JDBC**

- *eng. Java Database Connectivity*
- inicijalno razvijen od strane SUN Microsystems
 - 1996. godina
- standardni API za pristup bazama podataka u Java programskom jeziku
 - skup interfejsa koje proizvođač mora implementirati kako bi napravio drajver za svoju bazu podataka
 - sastavni deo JavaSE i JavaEE edicija
 - u paketu **java.sql.***
- nezavisan od proizvođača baze podataka
- nezavisan od operativnog sistema

JDBC

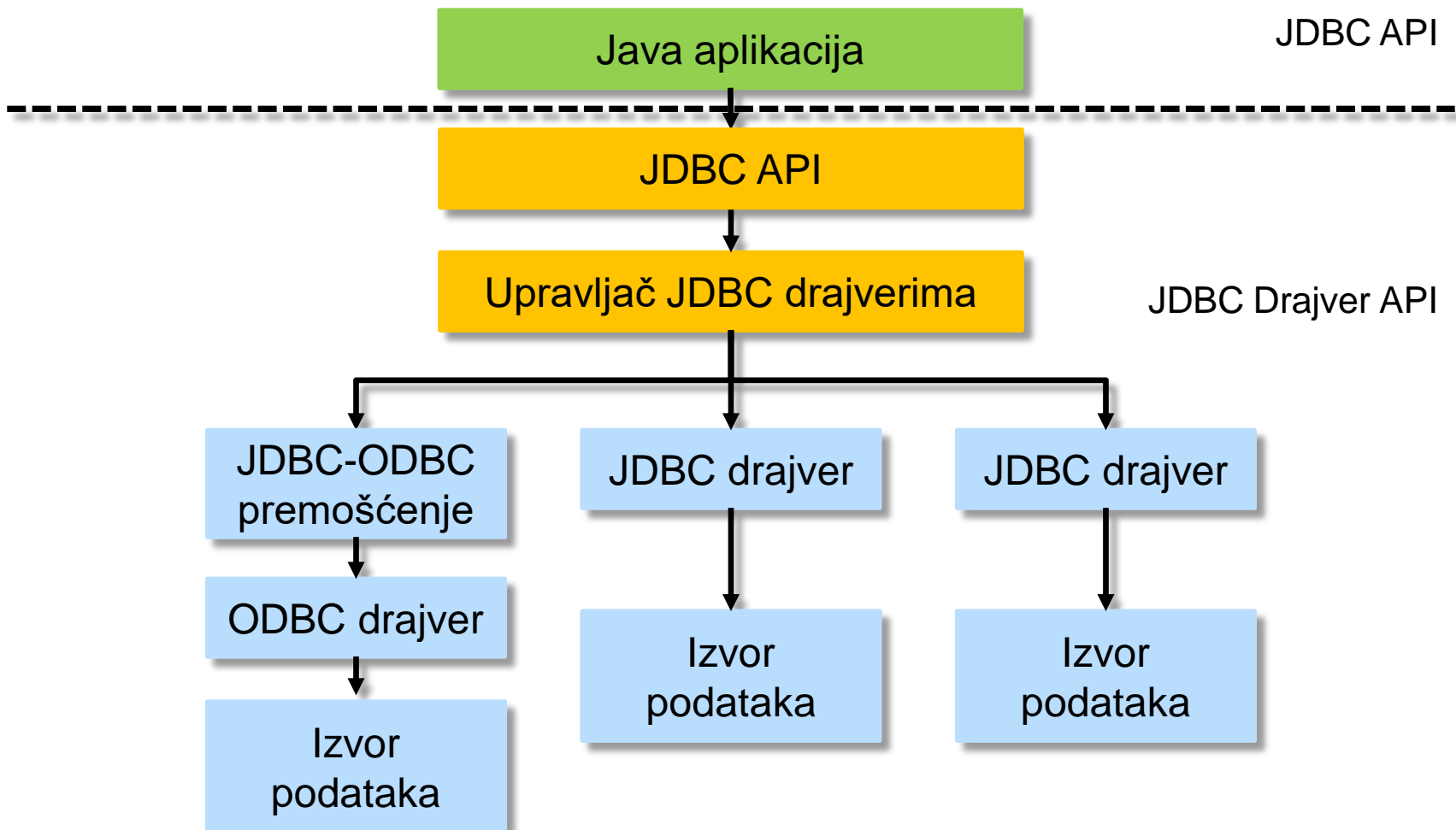
- **JDBC**
 - strukturalno i konceptualno vrlo sličan ODBC standardu
 - poboljšanja vezana za iskorišćenje svih mogućnosti Java programskog jezika

JDBC

- **Arhitektura JDBC sistema**
 - sastoji se iz dva glavna dela
 - **JDBC API**
 - obezbeđuje vezu između aplikacije i upravljača drajverima
 - **JDBC drajver API**
 - obezbeđuje vezu između upravljača drajverima i drajvera
 - JDBC koristi upravljač drajverima kako bi istovremeno podržao rad sa više od jedne baze podataka

ODBC

- Arhitektura JDBC sistema



JDBC

- **Arhitektura JDBC sistema**

- modeli arhitekture podržani od strane JDBC API-ja

- **dvoslojni model**

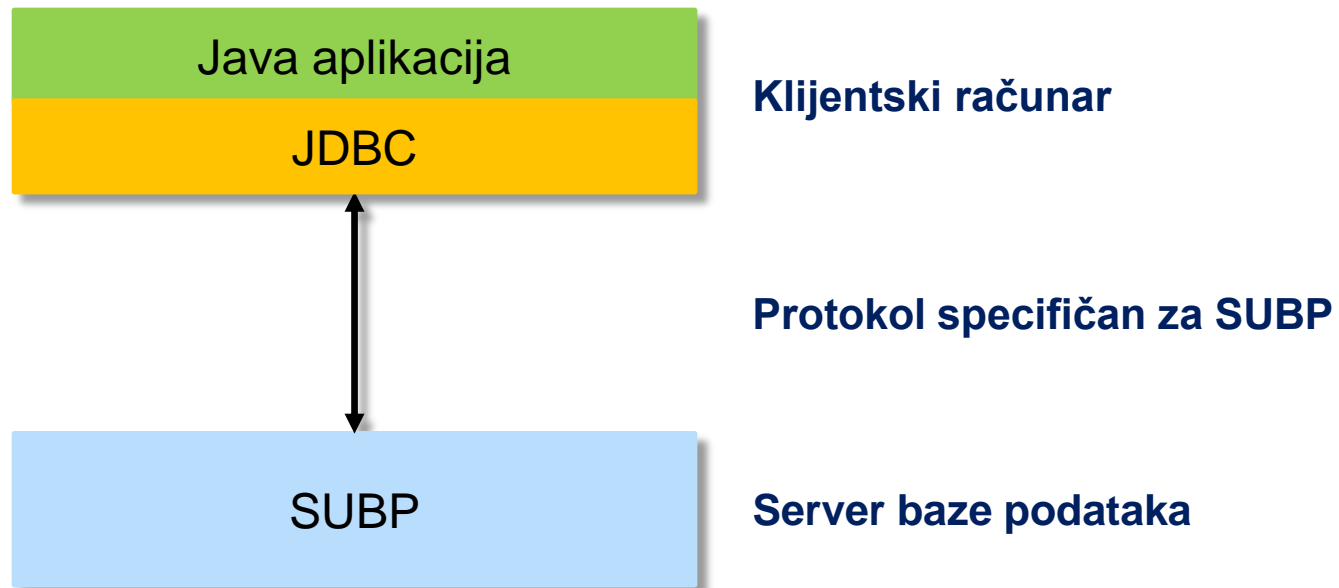
- aplikacija komunicira **direktno** sa izvorom podataka
 - » koji može biti na istom ili na drugom računaru

- **troslojni model**

- aplikacija komunicira **indirektno** sa izvorom podataka
 - » postoji međusloj koji nudi servise
 - » kontrole pristupa
 - » filtriranja sadržaja
 - » pojednostavljuje instalaciju aplikacija
 - » u dosta slučajeva poboljšava performanse

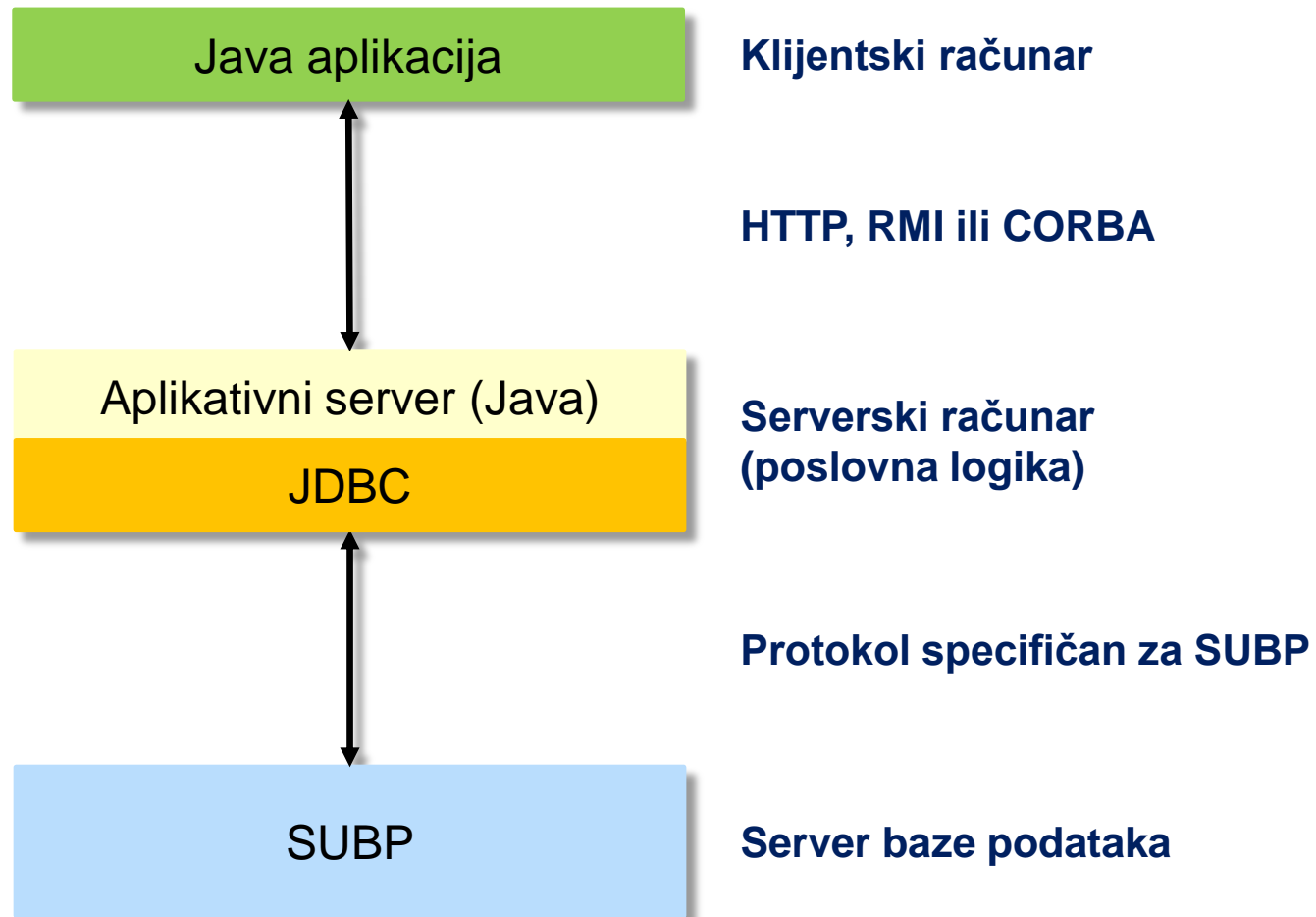
ODBC

- **Dvoslojni JDBC model**



ODBC

- Troslojni JDBC model



JDBC

- **Arhitektura JDBC sistema**

- tipovi drajvera

- **ODBC drajver sa JDBC-ODBC premošćenjem**

- *eng. The JDBC-ODBC Bridge plus ODBC driver*
 - koristi rasprostranjene ODBC drajvere
 - konvertuje JDBC pozive u ODBC pozive i prosleđuje ih ODBC drajveru

- **Java drajver sa direktnim pozivom API-ja BP**

- *eng. Native-API partly-Java driver*
 - drajver za dvoslojni model
 - konvertuje JDBC pozive u pozive nad API-jem baze podataka
 - koristi biblioteke koje se isporučuju sa svakim SUBP-om
 - » nema potrebe za premošćenjem
 - pruža pristup svim funkcionalnostima BP
 - » potrebno je koristiti JNI za poziv funkcija napisanih u C/C++

JDBC

- **Arhitektura JDBC sistema**

- tipovi drajvera

- **Java drajver sa nezavisnim međuslojem**

- *eng. JDBC-Net pure Java driver*
 - drajver za troslojni model
 - konvertuje JDBC pozive u protokol nezavisan od SUBP-a
 - » koji se konvertuje u SUBP-ov protokol na serverskoj strani
 - najfleksibilnija vrsta drajvera

- **Java drajver sa direktnom komunikacijom sa BP**

- *eng. Native-protocol pure Java driver*
 - konvertuje JDBC pozive u protokol koga koristi SUBP
 - » ne poziva funkcije u bibliotekama SUBP-a već direktno komunicira putem protokola

JDBC

- **Primer upotrebe JDBC sistema**
 - osnovni koraci u komunikaciji sa BP
 1. učitavanje drajvera
 2. definisanje URL-a veze
 3. uspostavljanje veze
 4. kreiranje izraza
 5. izvršavanje upita ili ažuriranja
 6. obrada rezultata
 7. zatvaranje veze

JDBC

- **Primer upotrebe JDBC sistema**

- osnovni koraci u komunikaciji sa BP

- 1. učitavanje drajvera**

- potrebno je učitati odgovarajuću klasu koja predstavlja drajver
 - » klasa mora biti u CLASSPATH-u
 - kreira se instanca drajvera
 - » registruje se u upravljaču drajverima

```
try {  
    Class.forName("connect.microsoft.MicrosoftDriver");  
    Class.forName("oracle.jdbc.driver.OracleDriver");  
    Class.forName("com.sybase.jdbc.SybDriver");  
} catch(ClassNotFoundException cnfe) {  
    System.err.println("Greska pri učitavanju drajvera: " + cnfe);  
}
```

JDBC

- **Primer upotrebe JDBC sistema**

- osnovni koraci u komunikaciji sa BP

- 2. definisanje URL-a veze**

- koriste JDBC protokol i poseduju
 - » naziv servera
 - » port
 - » naziv baze podataka
 - tačan format zavisi i od proizvođača drajvera

```
String host = "dbhost.imekompanije.com";
```

```
String dbName = „imeBP“;
```

```
int port = 1234;
```

```
String oracleURL = "jdbc:oracle:thin:@" + host + ":" + port + ":" + dbName;
```

```
String sybaseURL = "jdbc:sybase:Tds:" + host + ":" + port + ":" +  
                    "?SERVICENAME=" + dbName;
```


JDBC

- **Primer upotrebe JDBC sistema**

- osnovni koraci u komunikaciji sa BP

- 3. uspostavljanje veze**

- potrebno je proslediti

- » URL

- » korisničko ime

- » lozinku

```
String username = "username";
```

```
String password = "secret";
```

```
Connection connection = DriverManager.getConnection(oracleURL,  
username, password);
```

JDBC

- **Primer upotrebe JDBC sistema**

- osnovni koraci u komunikaciji sa BP

- 3. uspostavljanje veze**

- kreira se objekat klase *Connection*
 - moguće je dobiti informacije o bazi podataka

```
DatabaseMetaData dbMetaData = connection.getMetaData();  
String productName = dbMetaData.getDatabaseProductName();  
System.out.println("Database: " + productName);  
String productVersion = dbMetaData.getDatabaseProductVersion();  
System.out.println("Version: " + productVersion);
```

JDBC

- **Primer upotrebe JDBC sistema**

- osnovni koraci u komunikaciji sa BP

- 4. kreiranje izraza**

- služi za slanje naredbi bazi podataka

- tipovi izraza

- » **jednostavni izraz**

- » izvršava jednostavne SQL upite bez parametara

- » klasa *Statement*

- » **pripremljeni izraz**

- » izvršava unapred kompajlirane SQL upite sa ili bez parametara

- » klasa *PreparedStatement*

- » **pozivajući izraz**

- » izvršava poziv skladištene procedure iz baze podatka

- » klasa *CallableStatement*

JDBC

- **Primer upotrebe JDBC sistema**

- osnovni koraci u komunikaciji sa BP

- 4. kreiranje izraza**

```
//Statement
```

```
Statement statement = connection.createStatement();
```

```
//PreparedStatement
```

```
String updateString = „UPDATE " + dbName + ".Customers " +  
                        „SET ContactName = ? WHERE CustomerID = ?";
```

```
PreparedStatement pStatement = connection.prepareStatement(updateString);
```

```
//CallableStatement
```

```
String insertStoreProc = "{call deleteCustomer(?)}";
```

```
CallableStatement cStatement = dbConnection.prepareCall(insertStoreProc);
```

JDBC

- **Primer upotrebe JDBC sistema**

- osnovni koraci u komunikaciji sa BP

- 5. izvršavanje upita ili ažuriranja**

- nakon kreiranja izraza, naredba može da se izvrši

```
//Statement
```

```
String query = "SELECT CustomerID, ContactName FROM Customers";
```

```
ResultSet resultSet = statement.executeQuery(query);
```

```
//PreparedStatement
```

```
pStatement.setString(1, "Milan");
```

```
pStatement.setInt(2, "42");
```

```
int rowsAffected = pStatement.executeUpdate();
```

```
//CallableStatement
```

```
cStatement.setInt(1, "42");
```

```
int rowsAffected = cStatement.executeUpdate();
```

JDBC

- **Primer upotrebe JDBC sistema**

- osnovni koraci u komunikaciji sa BP

- 6. obrada rezultata**

- najjednostavnije je da se rezultati obrade red po red
 - kolone unutar reda su numerisane od broja 1
 - postoje pomoćne metode za pronalaženje kolona po imenu kao i za ispitivanje svojstava kolona

```
while(resultSet.next()) {  
    System.out.println( results.getString(1) + " " +  
                        results.getString(2) + " " +  
                        results.getString(3));  
}
```

JDBC

- **Primer upotrebe JDBC sistema**

- osnovni koraci u komunikaciji sa BP

- 7. zatvaranje veze**

- eksplicitno zatvaranje veze

- zatvaranje veze treba odgoditi do trenutka kada je baš neophodno

- » ponovno uspostavljanje veze zahteva mnogo vremena

```
connection.close();
```

Sadržaj

- Protokoli
- ODBC
- JDBC
- OLE DB
- ADO .NET
- Objektno-relaciono prevođenje
- Hibernate
- Entity Framework

OLE DB

- **Motivacija**

- postoji potreba za skladištenjem pojedinih podataka izvan baze podataka
- **problem**
 - ne postoji jedinstven način za pristup ovakvim podacima
 - nije moguće vršiti SQL upite nad ovakvim izvorima podataka
- **cilj**
 - omogućavanje jedinstvenog načina za pristup svim podacima
 - i unutar i izvan relacione baze podatka

OLE DB

- **OLE DB**

- *eng. Object Linking and Embedding, Database*
- inicijalno razvijen od strane Microsofta
 - kao naslednik i zamena za ODBC
 - uvedena podrška za pristup nerelacionim izvorima podataka
 - » koji, u opštem slučaju, ne podržavaju SQL

- **OLE DB API**

- standardni API koji nudi skup COM interfejsa
 - COM – *eng. Component Object Model*
 - za pristup podacima iz različitih izvora podataka
 - za definisanje dodatnih servisa baza podataka
- nezavisan od proizvođača baze podataka

OLE DB

- **Component Object Model (COM)**
 - sistem za kreiranje binarnih softverskih komponenti
 - nezavisan od platforme
 - distribuiran
 - objektno-orijentisan
 - postoji za veliki broj programskih jezika
 - osnova za OLE, ActiveX, itd.

OLE DB

- **Osnovne osobine OLE DB**

- **kreiranje specijalizovanih interfejsa**

- *eng. Interface Factoring*
 - namensko uvođenje novih interfejsa za aplikaciju
 - npr. formiranje interfejsa za čitanje, interfejsa za pisanje, itd.
 - nad jednim OLE objektom mogu biti primenljive različite operacije iz više interfejsa
 - *QueryInterface* metoda omogućava prikaz interfejsa
 - nad objektom su uvek primenljive sve operacije podržanog interfejsa
 - nije moguće ograničiti se na primenu samo određenih metoda interfejsa
 - nije moguće dinamički dodavati nove metode u interfejs

OLE DB

- **Osnovne osobine OLE DB**

- **odabir interfejsa**

- akcija koja omogućava dinamički odabir interfejsa koji je primenjen na neki objekat
 - kada prilikom pisanja aplikacije ne znamo da li objekat sadrži operacije nekog interfejsa
 - *IUnknown* interfejs je primenljiv na sve objekte
 - svi ostali interfejsi nasleđuju ovaj interfejs
 - sadrži operacije
 - » *QueryInterface*
 - » *AddRef*
 - » *Release*

OLE DB

- **Osnovne osobine OLE DB**

- **odabir interfejsa**

- operacije *IUnknown* interfejsa
 - *QueryInterface*
 - » operacija kojom se odabira interfejs pomoću GUID-a
 - » eng. *globally unique identifier*
 - » GUID je poznat za vreme kompajliranja
 - » vraća pokazivač na traženi interfejs ukoliko je na objektu primenjen taj interfejs
 - *AddRef*
 - » operacija kojom se uvećava broj referenci na interfejs objekta
 - *Release*
 - » operacija kojom se oslobađa referenca sa interfejsa objekta

OLE DB

- **Osnovne osobine OLE DB**

- **brojanje referenci**

- akcija utvrđivanja ukupnog broja referenci na interfejs nekog objekta
 - broj referenci se čuva u svakoj instanci pokazivača na interfejs
 - garantuje da objekat neće biti uništen sve dok postoje reference na njega

- **podrška unicode standarda**

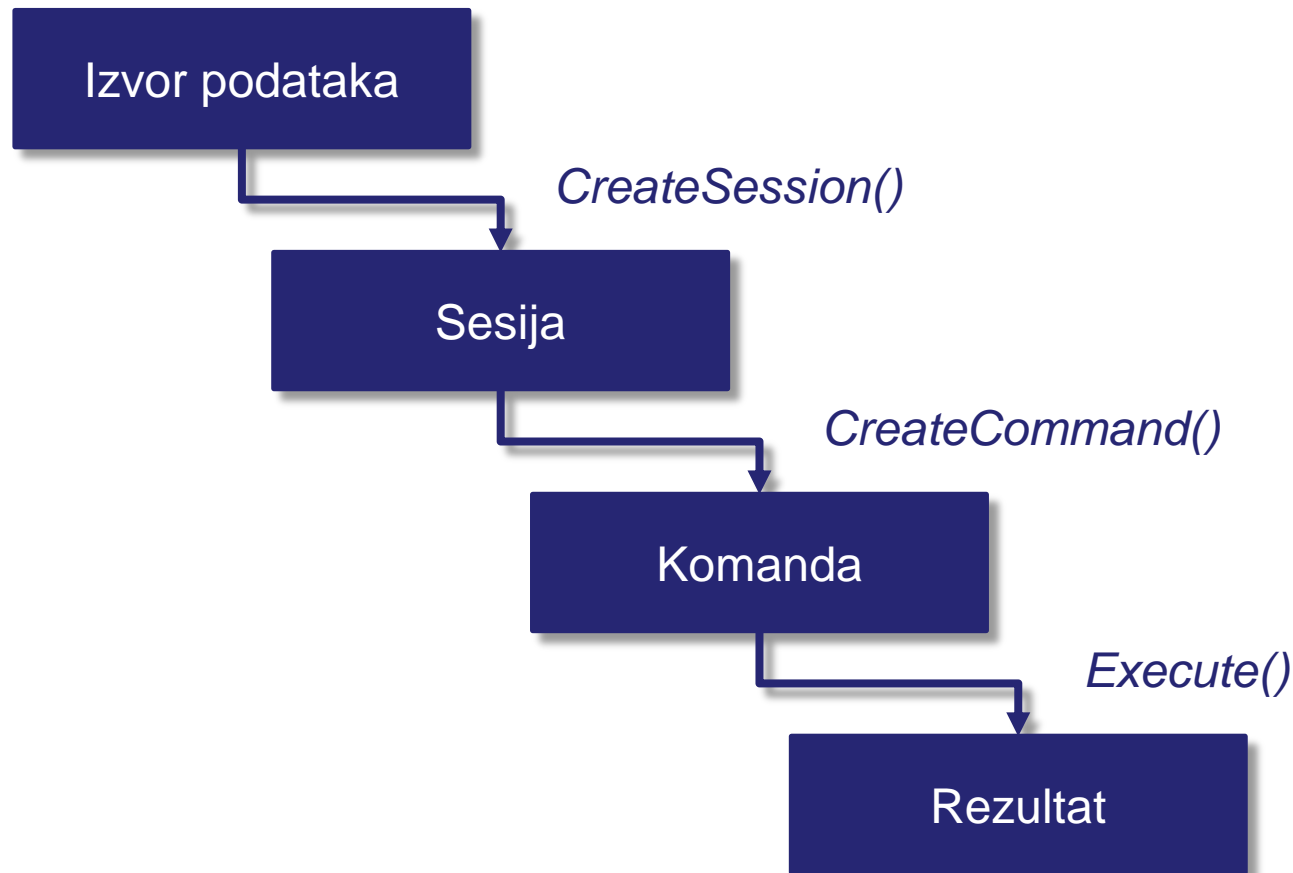
- garantuje da OLE DB rukuje *Unicode* stringovima
 - COM interfejsi rukuju *Unicode* stringovima
 - » umesto ANSI stringovima
 - izuzetak su dobavljanje i postavljanje stringova u tabelama koje sadrže ANSI podatke

OLE DB

- **Osnovne osobine OLE DB**
 - **upravljanje memorijom**
 - OLE koristi dve vrste memorije
 - lokalna aplikativna memorija
 - deljena memorija
 - zauzimanje memorije se obavlja iz **pozivanog** okruženja
 - oslobađanje memorije se obavlja iz **pozivajućeg** okruženja

OLE DB

- Osnovni objekti u OLE DB



OLE DB

- **Osnovni objekti u OLE DB**

- **objekat izvora podataka**

- objekat koji predstavlja izvor podataka
 - enkapsulira funkcionalnosti ODBC okruženja
 - vezu ka SUBP-u
 - informacije o vezi

- **objekat sesije**

- definiše opseg transakcije
 - generiše rezultat iz izvora podataka
 - generiše objekte komandi koje predstavljaju SQL komande
 - upiti, DDL, DML, itd.
 - date SQL komande moraju biti podržane od strane ciljnog izvora podataka

OLE DB

- **Osnovni objekti u OLE DB**

- **objekat komande**

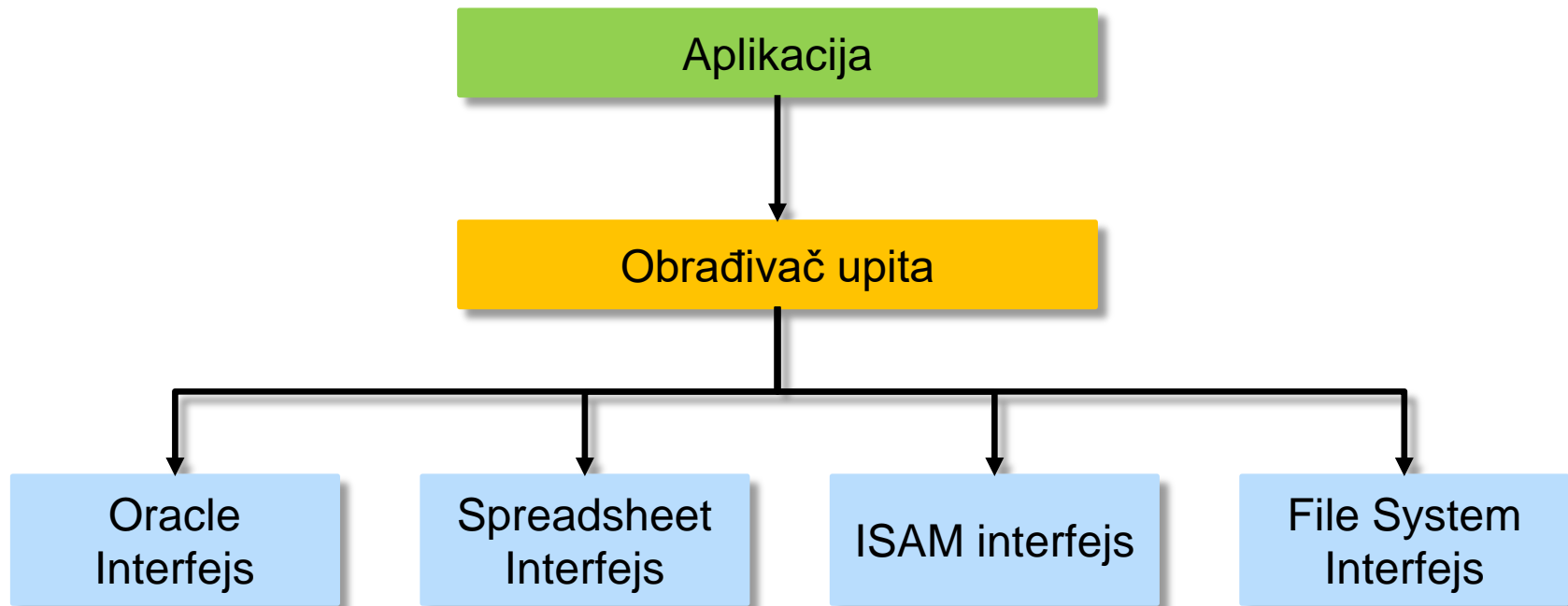
- objekat koji predstavlja komandu koja se upućuje izvoru podataka
 - ukoliko izvor podataka podržava komandu datog tipa
 - generiše ga objekat sesije
 - enkapsulira funkcionalnosti ODBC izraza u neizvršenom stanju
 - obezbeđuje prosleđivanje parametara

- **objekat rezultata**

- objekat koji predstavlja rezultat upita
 - nakon izvršenja komande

OLE DB

- Arhitektura OLE DB sistema



OLE DB

- **Arhitektura OLE DB sistema**
 - zasniva se na upotrebi servisa COM infrastrukture, čime se
 - smanjuje broj duplih servisa
 - povećava interoperabilnost
 - između različitih izvora podataka
 - između različitih alata i okruženja
 - enkapsulira deo funkcionalnosti SUBP
 - pristup podacima i njihovo ažuriranje
 - korišćenje informacija iz rečnika BP
 - izvršavanje transakcija
 - obezbeđenje sigurnosti
 - udaljeni pristup podacima

OLE DB

- **Osnovni koraci korišćenja OLE DB**
 1. inicijalizacija OLE-a
 2. povezivanje sa izvorom podataka
 3. kreiranje i izvršavanje naredbe
 4. obrada rezultata
 5. oslobađanje objekata i deinicijalizacija OLE-a

Sadržaj

- Protokoli
- ODBC
- JDBC
- OLE DB
- ADO .NET
- Objektno-relaciono prevođenje
- Hibernate
- Entity Framework

ADO .NET

- **Motivacija**
 - razvoj Internet aplikacija
 - zahteva nov pristup čuvanju i dobijanju informacija
 - sve veći značaj XML-a u komunikaciji
 - .NET okruženje nema standardni mehanizam za pristup podacima

ADO .NET

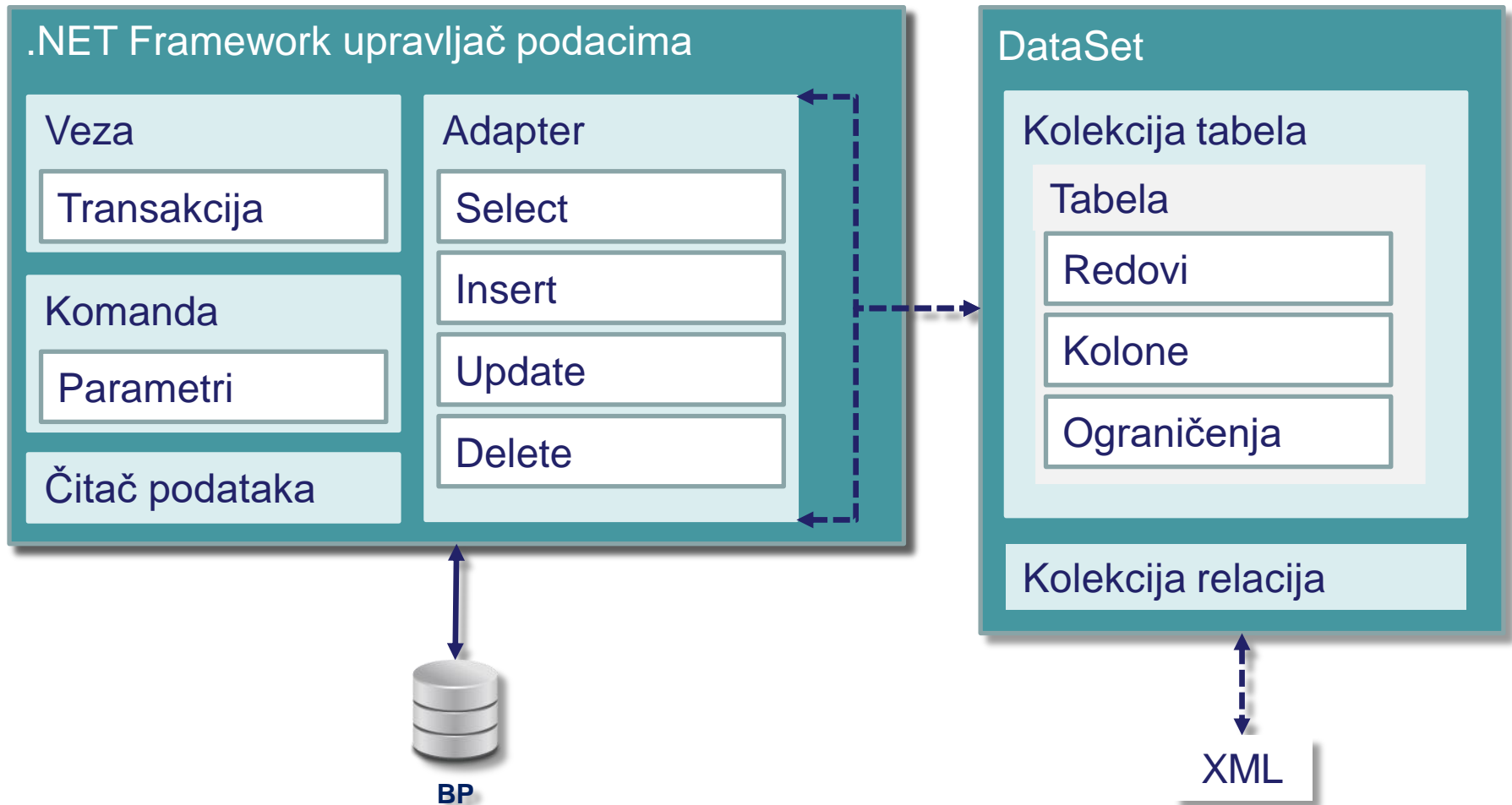
- **ADO .NET**
 - *eng. ActiveX Data Objects, .NET*
 - pruža konzistentan način za pristup izvorima podataka
 - moguće je koristiti OLE DB ili ODBC kao posrednike u tom pristupu
 - deo je osnovne biblioteke uključene u .NET okruženje

ADO .NET

- **ADO .NET**
 - ciljevi razvoja
 - *ActiveX Data Objects* (ADO)
 - skup COM objekata koji služi za pristup izvorima podataka
 - koristi se kako posrednik u komunikaciji sa OLE DB
 - ADO .NET izmenjuje ADO u velikoj meri
 - ADO .NET se može posmatrati kao sasvim nov proizvod
 - pruža podršku za višeslojne aplikacije
 - podrška za *offline* obradu podataka
 - » u kojoj se učitani podaci obrađuju nakon zatvaranja veze sa bazom podataka
 - poseduje integrisanu podršku za pristup i obradu podataka u XML formatu

ADO .NET

- Arhitektura ADO .NET sistema



ADO .NET

- **Arhitektura ADO .NET sistema**

- **upravljač podacima**

- *eng. data provider*
 - komponenta za upravljanje podacima i čitanje podataka
 - objekti
 - veza
 - komanda
 - čitač podataka
 - » *eng. DataReader*
 - » tok podataka od izvora
 - adapter za podatke
 - » *eng. DataAdapter*
 - » premošćenje između izvora i *DataSet-a*

ADO .NET

- **Arhitektura ADO .NET sistema**

- **DataSet**

- skup objekata namenjen za pristup podacima, koji je nezavisan od izvora podataka
 - sadrži objekte organizovane u obliku tabela
 - *eng. DataTable*
 - redovi i kolone
 - primarni i strani ključ
 - ograničenja
 - metapodaci

ADO .NET

- **Arhitektura ADO .NET sistema**

- podržava izbor načina pristupa podacima
 - putem čitača podataka ili putem DataSet-a
 - zavisi od funkcionalnosti koje zahteva aplikacija
 - čitač podataka
 - služi isključivo za čitanje podataka
 - mnogo bolje performanse prilikom čitanja od DataSet-a
 - DataSet
 - predstavlja lokalni *cache* podataka aplikacije
 - omogućava udaljeni pristup podacima
 - omogućava dinamičku interakciju sa podacima
 - » Windows Forme
 - » kombinacija različitih izvora podataka
 - pogodan za tzv. intenzivnu obradu podataka
 - » bez višestrukog uspostavljanja veze sa izvorom podataka

ADO .NET

- **ADO .NET i XML**
 - DataSet može biti popunjen direktno iz XML dokumenta
 - uključujući i informacije o šemi
 - XML dokument može biti formiran iz DataSet-a
 - moguće je i kreiranje šeme ukoliko DataSet sadrži potrebne informacije
 - kako bi omogućio prenos podataka putem HTTP-a

ADO .NET

- **Primer upotrebe ADO .NET**

```
using System;  
using System.Data;  
using System.Data.SqlClient;
```

```
class Program
```

```
{
```

```
static void Main()
```

```
{
```

```
    //Veza sa MSSQL bazom podataka
```

```
    string connectionString = "Data Source=(local);Initial Catalog=Northwind;" +  
    "Integrated Security=true";
```

```
    // Upit sa jednim mestom za parametar
```

```
    string queryString =
```

```
    "SELECT ProductID, UnitPrice, ProductName from products "
```

```
    + "WHERE UnitPrice > @pricePoint "
```


ADO .NET

- **Primer upotrebe ADO .NET**

```
// specifikacija parametra upita
```

```
int paramValue = 5;
```

```
// Kreiranje veze u using bloku. Garantuje zatvaranje veze na kraju bloka.
```

```
using (SqlConnection connection = new SqlConnection(connectionString))
```

```
{
```

```
    // kreiranje komande i dodavanje parametra
```

```
    SqlCommand command = new SqlCommand(queryString, connection);
```

```
    command.Parameters.AddWithValue("@pricePoint", paramValue);
```

```
    try
```

```
    {
```

```
        connection.Open();
```

```
        SqlDataReader reader = command.ExecuteReader();
```

```
        while (reader.Read())
```

```
            Console.WriteLine("{0}\t{1}\t{2}", reader[0], reader[1], reader[2]);
```

ADO .NET

- **Primer upotrebe ADO .NET**

```
//zatvaranje čitača
reader.Close();
} //try blok
catch (Exception ex)
{
    Console.WriteLine(ex.Message);
}

Console.ReadLine();

} //using blok
} //main funkcija
```

Sadržaj

- Protokoli
- ODBC
- JDBC
- OLE DB
- ADO .NET
- Objektno-relaciono prevođenje
- Hibernate
- Entity Framework

Objektno-relaciono prevođenje

- **Objektno-relaciono prevođenje (ORM)**
 - primenjuje se u objektno orijentisanim programskim jezicima
 - kreira sliku virtuelne objektne baze podataka
 - koja se koristi u aplikaciji
 - rešava konceptualne i tehničke probleme
 - koji nastaju zbog potrebe korišćenja relacionih baza podataka u objektno orijentisanim aplikacijama

Objektno-relaciono prevođenje

- **Objektno-relaciono prevođenje (ORM)**
 - objektno-relaciono mapiranje (preslikavanje)
 - *eng. object-relational mapping*
 - tehnika transformacije nekompatibilnih sistema
 - tipizacije koncepata
 - objektno-orijentisanog i relacionog modela
 - *eng. object-relational impedance mismatch*

Objektno-relaciono prevođenje

- **Motivacija**

- **problemi**

- problem nepostojanja ekvivalentnog relacionog koncepta u odnosu na neke objektno orijentisane koncepte
 - enkapsulacija
 - » skrivanje podataka
 - » prevođenje takvih podataka u relacionu BP može da naruši kvalitet podataka
 - modifikatori pristupa
 - » private, public, protected, itd.
 - » nisu apsolutne karakteristike podataka
 - » u zavisnosti od upotrebe, podaci mogu imati različite modifikatore
 - polimorfizam, nasleđivanje, interfejsi
 - » nisu u osnovi podržani od strane relacionog modela

Objektno-relaciono prevođenje

- **Motivacija**

- **problemi**

- problem prevazilaženja razlika u tipovima podataka
 - relacioni model striktno zabranjuje pokazivače
 - skalarni tipovi takođe mogu da se razlikuju značajno
 - » ograničeni znakovni tip u relacionom modelu
 - problem prevazilaženja strukturalne i integritetne razlike
 - ugnježdavanje objekata do proizvoljnog stepena
 - » relacije se ne ugnježdavaju
 - ograničenja u OO jezicima se vrlo retko eksplicitno definišu
 - » kroz uslovne izraze i mehanizam izuzetaka mogu da se definišu ograničenja u samom kôdu
 - problem prevazilaženja razlike pri manipulaciji
 - deklarativnost u relacionom modelu
 - imperativnost u OO modelu

Objektno-relaciono prevođenje

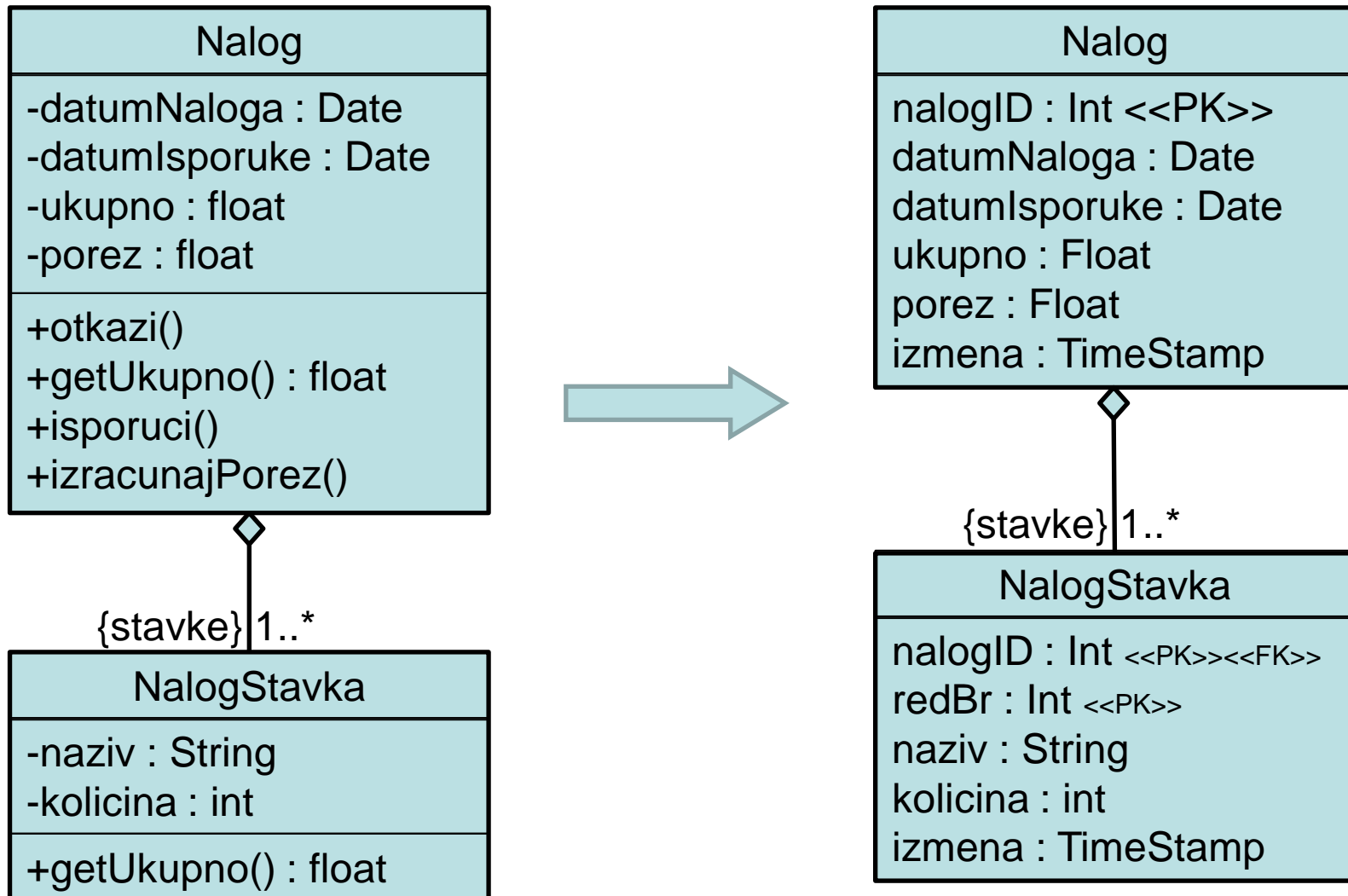
- **Osnovni ORM koncepti**
 - prevođenje klasa
 - često se jedna klasa prevodi u jednu tabelu
 - ne mora uvek biti slučaj
 - » npr. kod nasleđivanja
 - potrebna informacija o primarnom ključu
 - potrebne informacije o vezama sa drugim objektima
 - zbog formiranja stranih ključeva
 - prevođenje atributa klasa
 - prevode se u kolone tabele
 - ukoliko su skalarne vrednosti
 - kompleksnije prevođenje za attribute – objekte
 - ne prevode se atributi koji se koriste za privremeno smeštanje rezultata izračunavanja

Objektno-relaciono prevođenje

- **Osnovni ORM koncepti**
 - prateće informacije
 - *eng. shadow information*
 - sve dodatne informacije koje su potrebne za skladištenje objekta
 - ne obuhvataju podatke iz domena
 - obuhvataju
 - informacije o primarnom ključu
 - » koji uobičajeno predstavlja veštački ključ
 - informacije potrebne za kontrolu konkurentnosti
 - » vremenski trenuci, brojevi verzija, brojači, itd.
 - oznake logičkog brisanja

Objektno-relaciono prevodenje

• Primer



Objektno-relaciono prevođenje

- **Osnovni ORM koncepti**

- nasleđivanje

- relacioni model ne podržava strukture koje reprezentuju nasleđivanje
 - potrebno je posebno voditi računa o prevođenju nasleđivanja
 - načini prevođenja nasleđivanja
 - **prevođenje kompletne hijerarhije u jednu tabelu**
 - **prevođenje svake konkretne klase u posebnu tabelu**
 - **prevođenje svake klase u posebnu tabelu**
 - **prevođenje klase u generičku strukturu tabela**
 - višestruko nasleđivanje
 - mehanizam isti kao i kod jednostrukog nasleđivanja

Objektno-relaciono prevođenje

- **Osnovni ORM koncepti**

- nasleđivanje

- **prevođenje kompletne hijerarhije u jednu tabelu**

- svi atributi se smeštaju u jednu tabelu

- potreban poseban način za identifikovanje tipa torke

- » jedna kolona

- » vrednost kolone definiše kojoj klasi objekat pripada

- » predstavlja problem kada imamo mnogo različitih tipova

- » više boolean kolona

- dobra strategija

- » kada nema mnogo potklasa

- » kada nema mnogo preklapanja između potklasa

Objektno-relaciono prevođenje

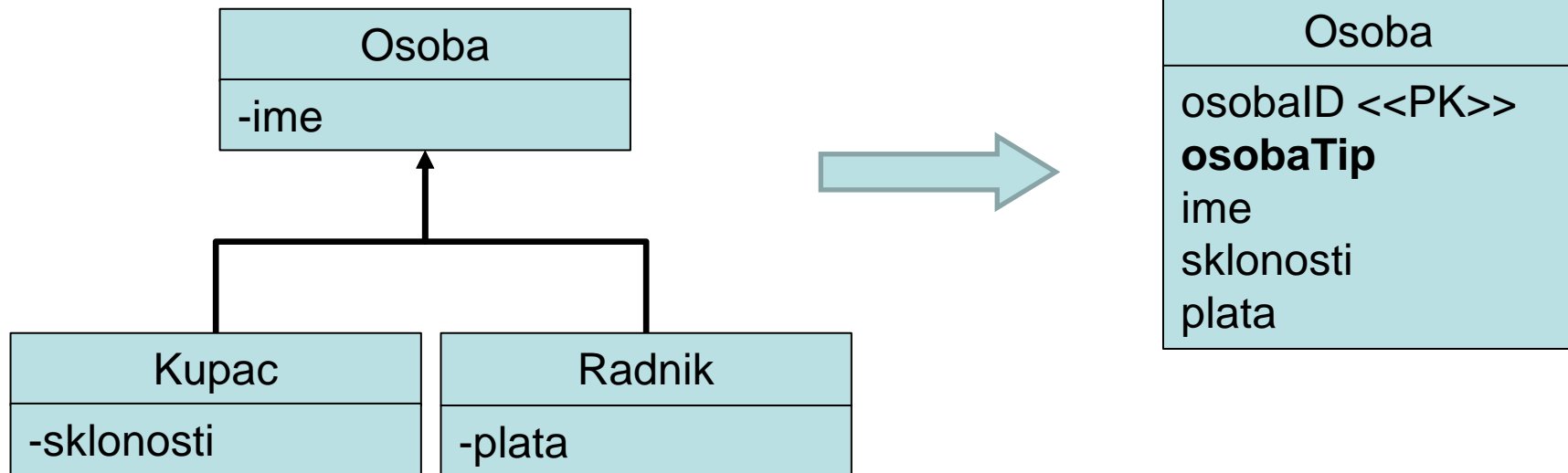
- **Osnovni ORM koncepti**
 - nasleđivanje
 - **prevođenje kompletne hijerarhije u jednu tabelu**
 - prednosti
 - » jednostavan pristup
 - » lako dodati nove klase
 - » dodavanjem kolona
 - » podržava polimorfizam
 - » promenom tipa
 - » brz pristup podacima

Objektno-relaciono prevođenje

- **Osnovni ORM koncepti**
 - nasleđivanje
 - **prevođenje kompletne hijerarhije u jednu tabelu**
 - mane
 - » promena strukture jedne klase menja strukturu svih klasa
 - » potencijalno loše iskorišćena memorija
 - » kompleksna vrednost dodatnog atributa koji predstavlja tip
 - » ukoliko ima preklapanja između potklasa
 - » sadržaj tabele može vrlo brzo biti uvećan za velike hijerarhije

Objektno-relaciono prevodenje

- Primer



Objektno-relaciono prevođenje

- **Osnovni ORM koncepti**
 - nasleđivanje
 - **prevođenje svake konkretne klase u posebnu tabelu**
 - tabela se kreira za svaku konkretnu klasu
 - » ne kreiraju se tabele za apstraktne klase
 - svaka tabela sadrži attribute klase kao i nasleđene attribute
 - dobra strategija
 - » kada nema mnogo preklapanja između potklasa

Objektno-relaciono prevođenje

- **Osnovni ORM koncepti**

- nasleđivanje

- **prevođenje svake konkretne klase u posebnu tabelu**

- prednosti

- » lak pristup podacima jednog objekta

- » nalaze se u jednoj tabeli

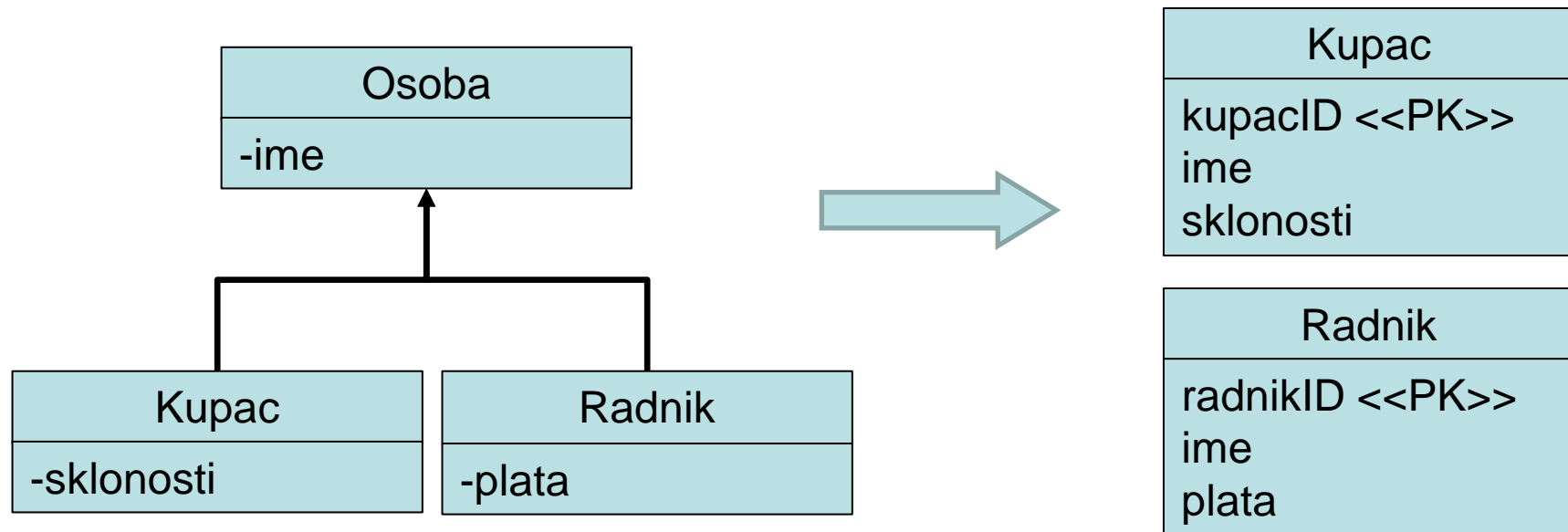
- mane

- » modifikacija apstraktne klase se propagira na sve njene potklase

- » teško napraviti tabele kada imamo preklapanje između potklasa

Objektno-relaciono prevodenje

- Primer



Objektno-relaciono prevođenje

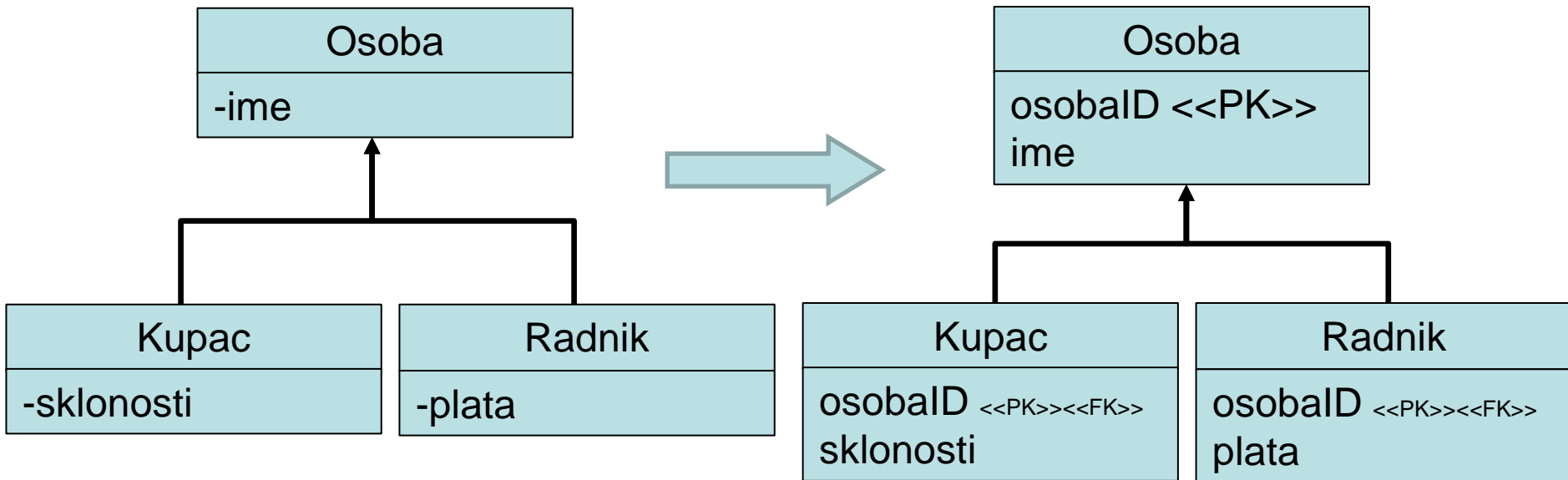
- **Osnovni ORM koncepti**
 - nasleđivanje
 - **prevođenje svake klase u posebnu tabelu**
 - tabela se kreira za svaku klasu
 - primarni ključ se nasleđuje od roditeljske klase
 - opciono dodavanje boolean kolone u roditeljsku
 - » kako bi se znao tip podatka bez pretrage potklasa
 - dobra strategija
 - » kada postoji preklapanje između potklasa
 - » podaci vrlo često menjaju tip

Objektno-relaciono prevođenje

- **Osnovni ORM koncepti**
 - nasleđivanje
 - **prevođenje svake klase u posebnu tabelu**
 - prednosti
 - » lako za razumevanje
 - » podržava polimorfizam
 - » laka modifikacija natklase i dodavanje novih potklasa
 - mane
 - » veliki broj tabela u BP
 - » potencijalno dugo čitanje
 - » zbog spajanja nekoliko tabela

Objektno-relaciono prevođenje

- Osnovni ORM koncepti



Objektno-relaciono prevođenje

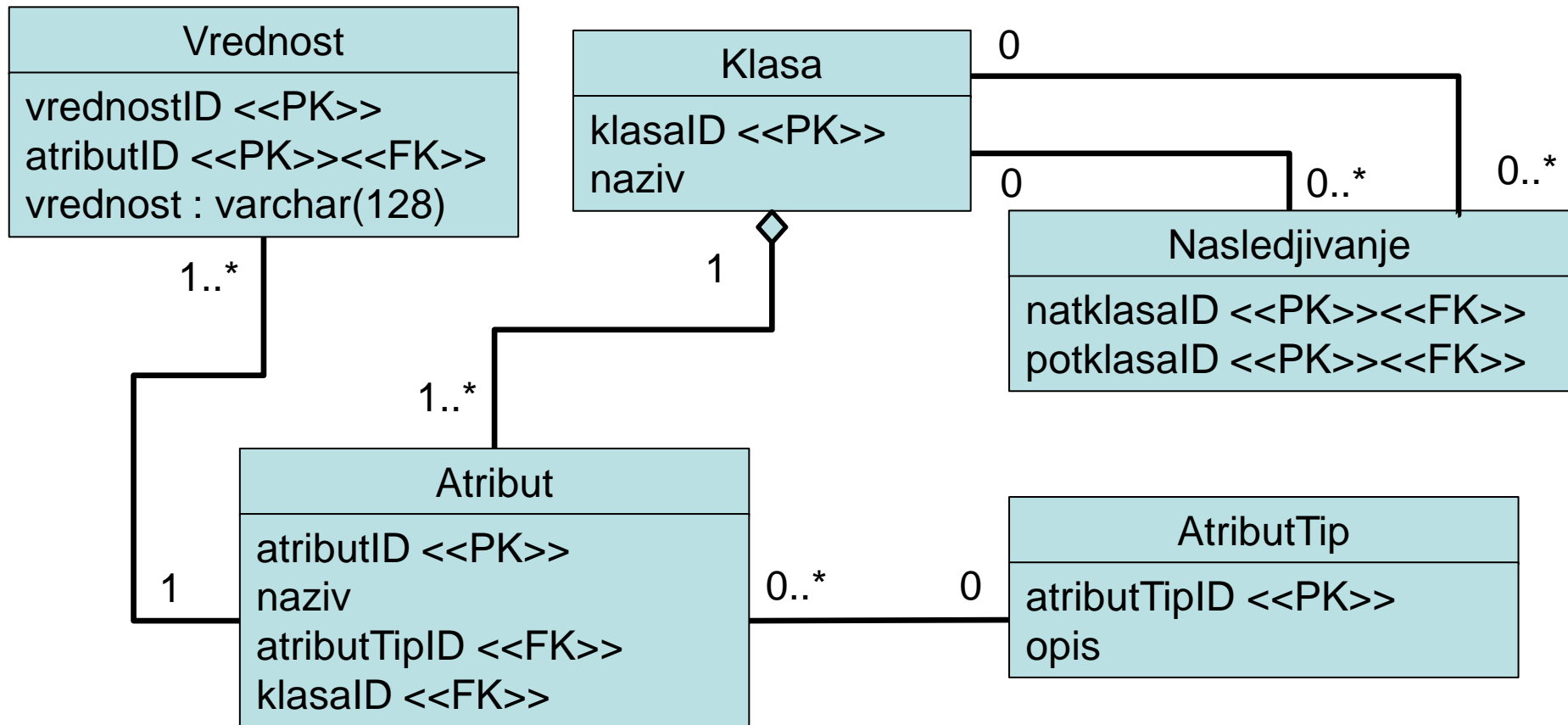
- **Osnovni ORM koncepti**
 - nasleđivanje
 - **prevođenje klase u generičku strukturu tabela**
 - pristup baziran na metapodacima
 - ovaj pristup nije ograničen samo na nasleđivanje
 - » podržava različite vrste prevođenja
 - dobra strategija za kompleksne aplikacije
 - » koje rade sa malim brojem podataka
 - » koje ne pristupaju podacima često

Objektno-relaciono prevođenje

- **Osnovni ORM koncepti**
 - nasleđivanje
 - **prevođenje klase u generičku strukturu tabela**
 - prednosti
 - » može biti proširen da podrži različite vrste prevođenja
 - » fleksibilno
 - » jednostavnom promenom atributa tabela
 - mane
 - » napredna tehnika koja nije jednostavna za implementaciju
 - » radi dobro samo za male količine podataka
 - » usled potrebe za spojevima nad više tabela

Objektno-relaciono prevođenje

- Primer



Objektno-relaciono prevodenje

- **Osnovni ORM koncepti**
 - prevodenje veza između objekata
 - podela veza po maksimalnim kardinalitetima
 - **veza jedan-na-jedan**
 - **veza jedan-na-više**
 - **veza više-na-više**
 - podela veza po vidljivosti
 - **jednosmerna**
 - » jedan povezani objekat „vidi“ drugi povezani objekat
 - » ne važi obrnuto
 - **dvosmerna**
 - » povezani objekti se „vide“ međusobno

Objektno-relaciono prevođenje

- **Osnovni ORM koncepti**
 - prevođenje veza između objekata
 - implementacija veze u objektima
 - maksimalan kardinalitet 0 ili 1
 - » referenca na povezani objekat
 - » get i set operacije
 - maksimalan kardinalitet N
 - » kolekcija povezanih objekata
 - » operacije za manipulisanje kolekcijom
 - jednosmerna
 - » samo je implementirana u objektu koji vidi povezani objekat
 - dvosmerna
 - » implementirana u oba objekta

Objektno-relaciono prevođenje

- **Osnovni ORM koncepti**

- prevođenje veza između objekata

- implementacija veze u relacionoj BP

- veze između entiteta se održavaju pomoću stranih ključeva

- veza jedan-na-jedan

- » strani ključ se nalazi u jednoj od tabela

- veza jedan-na-više

- » strani ključ se nalazi u tabeli na strani gde je maksimalni kardinalitet N

- » moguće napraviti posebnu tabelu

- veza više-na-više

- » posebna tabela

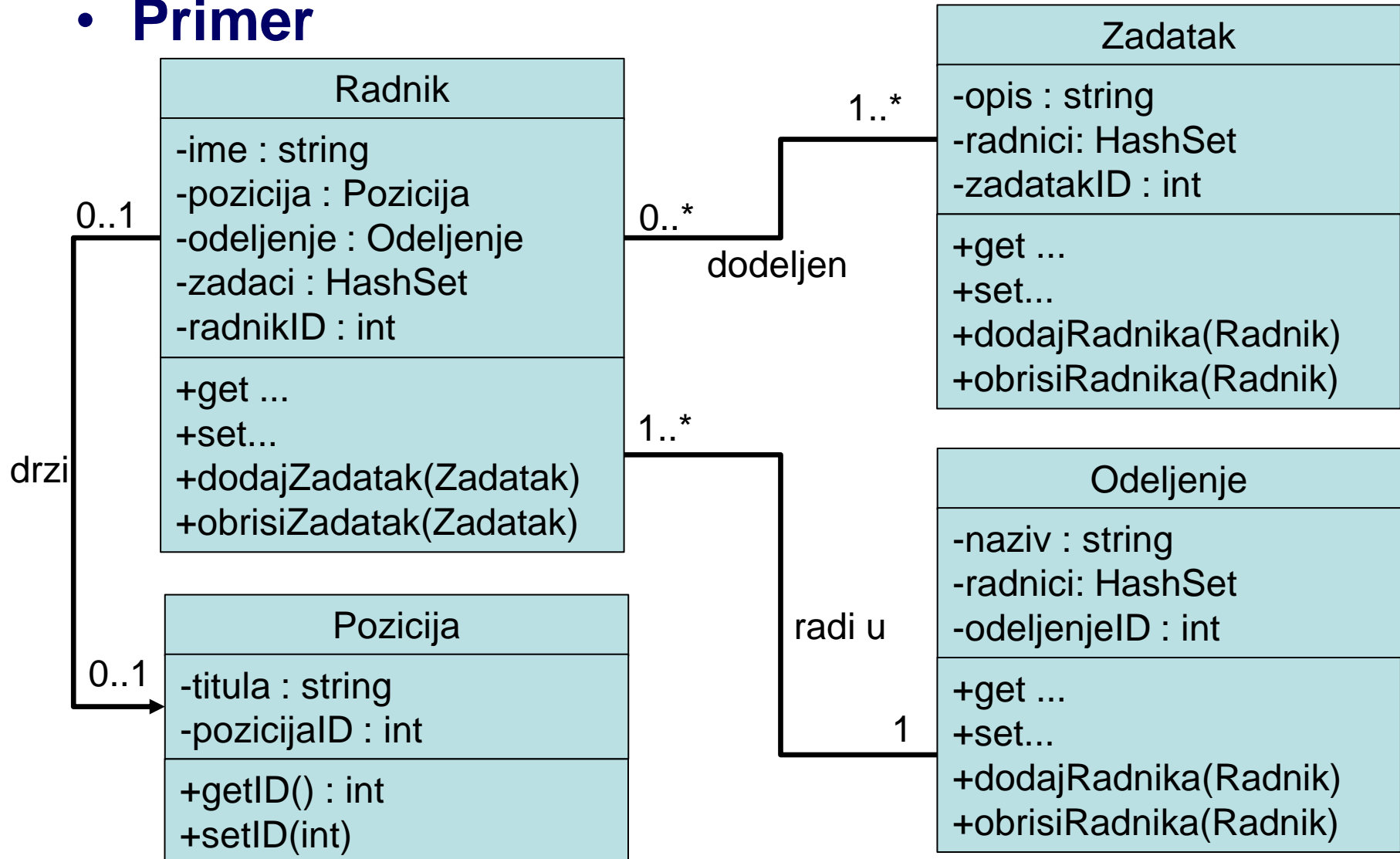
- » asocijativna tabela

- » sadrži primarne ključeve od obe tabele

- sve veze u relacionoj BP su dvosmerne

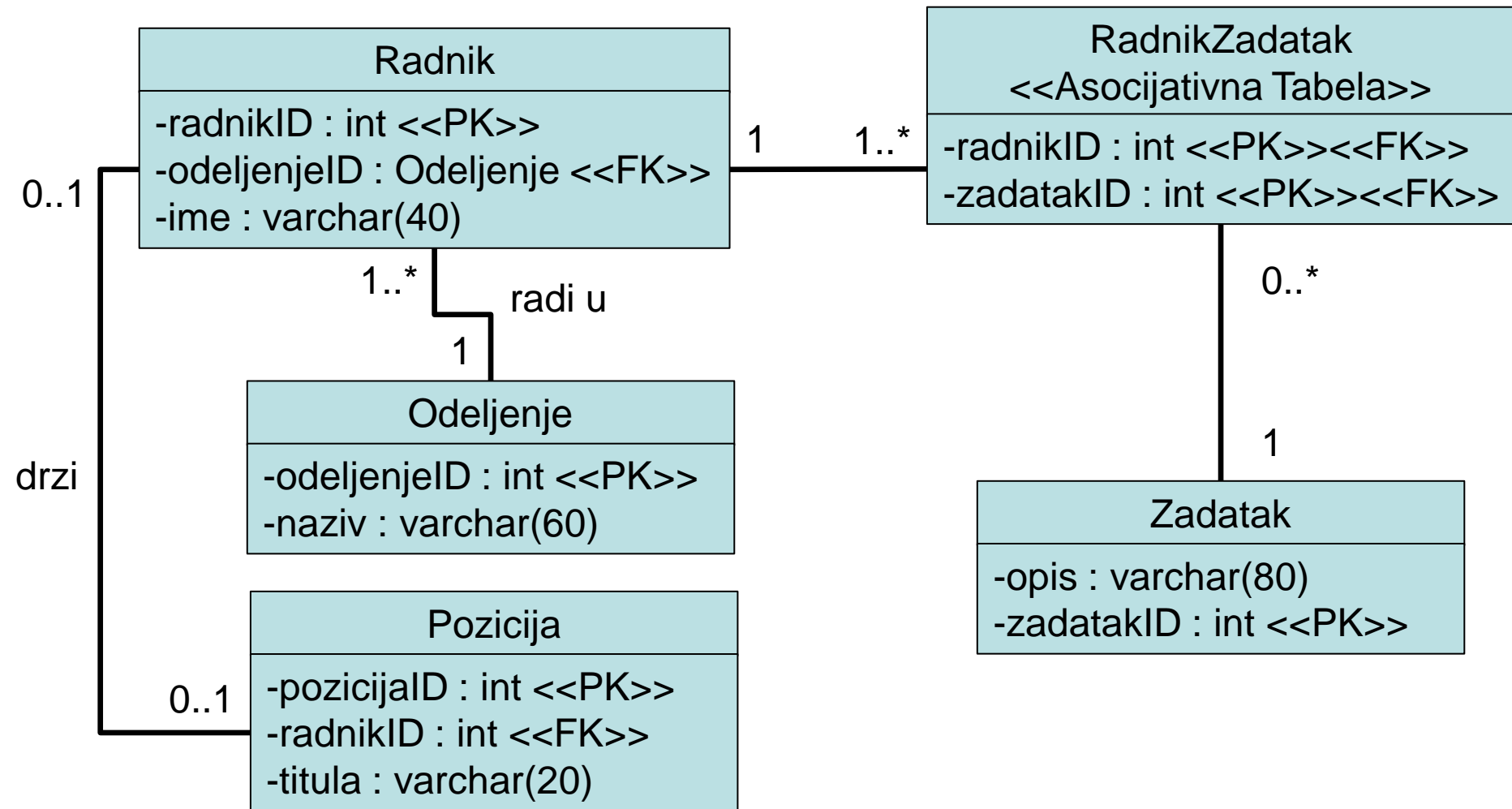
Objektno-relaciono prevođenje

• Primer



Objektno-relaciono prevođenje

• Primer



Objektno-relaciono prevođenje

- **Osnovni ORM koncepti**
 - prevođenje statičkih elemenata klase
 - strategije
 - **tabela sa jednim redom i jednom kolonom**
 - » jednostavna i brz pristup
 - **tabela sa jednim redom i više kolona za jednu klasu**
 - » svaka kolona za po jedno statičko polje
 - » jednostavna i brz pristup
 - **tabela sa jednim redom i više kolona za sve klase**
 - » minimalan broj tabela
 - » kolona za svako statičko polje
 - **generička tabela sa više redova za sve klase**

Sadržaj

- Protokoli
- ODBC
- JDBC
- OLE DB
- ADO .NET
- Objektno-relaciono prevođenje
- Hibernate
- Entity Framework

Hibernate

- **Hibernate**
 - ORM biblioteka za Java programski jezik
 - besplatna biblioteka
 - GNU licenca
 - ne zavisi od platforme
 - omogućava razrešenje problema korišćenja relacionih pojmova u objektno orijentisanim aplikacijama
 - u aplikacijama se koriste objekti umesto direktnog pristupa BP
 - poseduje svoj upitni jezik HQL
 - *eng. Hibernate Query Language*
 - NHibernate
 - verzija za C#

Hibernate

- **Hibernate**
 - koncepti ovog ORM-a se mogu naći u EJB3.0
 - *eng. Enterprise Java Beans 3.0*
 - u obliku binova i JPA
 - *eng. Java Persistence API*
 - Hibernate se može posmatrati kao implementacija *entity* binova
 - trenutni API je bogatiji od JPA

Hibernate

- **Hibernate**

- pre Java v5

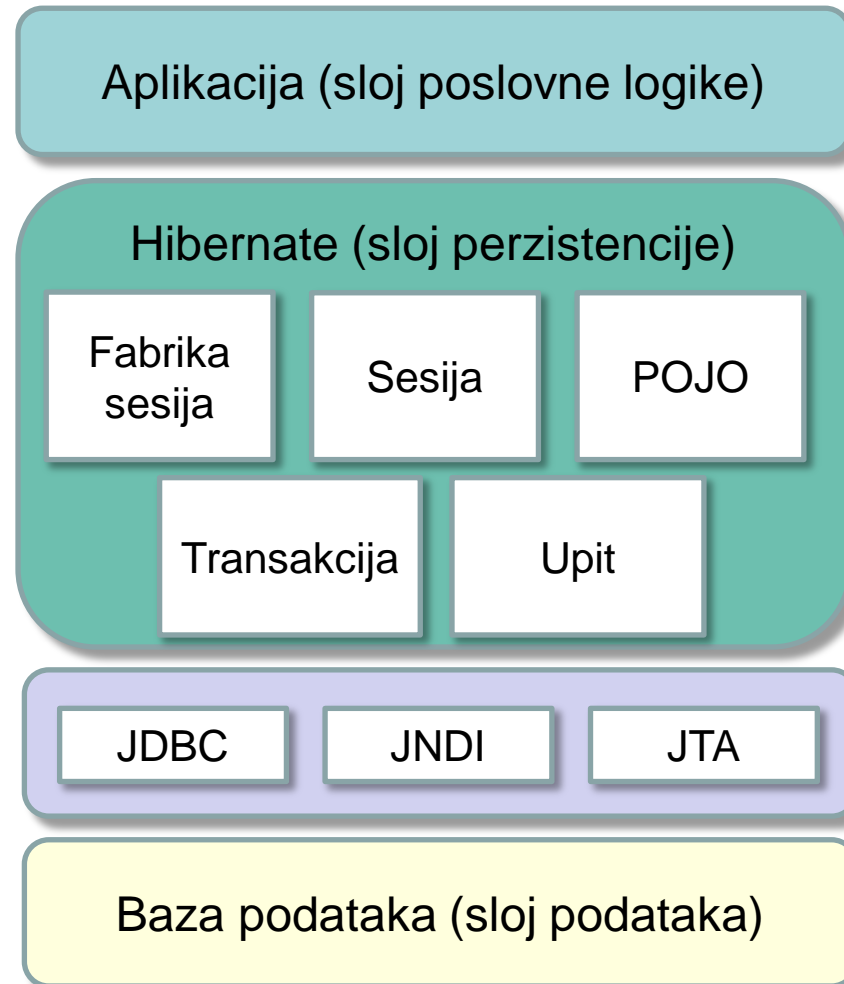
- konfigurisanje pravila prevođenja putem XML datoteke
 - složeno
 - odvojeno od kôda
 - » nije uočljivo prilikom pregledanja kôda
 - sporo povezivanje oba dela
 - još uvek podržan ali se retko koristi

- nakon Java v5

- mehanizam prevođenja putem **anotacija**
 - način da se podrže metapodaci
 - informacije kompajlerima ili okruženjima za izvršavanje

Hibernate

- **Arhitektura Hibernate-a**



Hibernate

- **Arhitektura Hibernate-a**

- **kreator sesija**

- *org.hibernate.SessionFactory*
 - dodeljuje sesije vezama iz skupa slobodnih veza
 - podrška za konkurentno kreiranje sesija

- **sesija**

- *org.hibernate.Session*
 - kanal komunikacije aplikacije i sistema baze podataka
 - jedna programska nit koja obuhvata razmenu podataka između aplikacije i baze podataka

- **transakcija**

- *org.hibernate.Transaction*
 - jedna programska nit koja predstavlja najmanju jedinicu obrade podataka

Hibernate

- **Arhitektura Hibernate-a**

- **upit**

- *org.hibernate.Query*
 - koristi se za kreiranje upita nad bazom podataka, putem
 - jezika HQL ili
 - nekog od dijalekata jezika SQL
 - realizuje se isključivo unutar sesije

Hibernate

- **Primer upotrebe Hibernate-a**

@Entity

@Table(name="Student")

```
public class Student {
```

```
    //anotacije mogu da stoje i ispred samih polja, ali se preferira da stoje uz njihove  
        get metode
```

```
    private Long id;
```

```
    protected String name;
```

```
    private AddressEntity address;
```

```
    // prevodimo primarni ključ čija vrednost je automatski generisana
```

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

@Column(name="STUDENT_ID")

```
    public Long getId() {
```

```
        return id;
```

```
    } ...
```

```
}
```

Hibernate

- **Primer upotrebe Hibernate-a**

@Entity

@Table(name="Student")

```
public class Student {  
    ...  
    public void setId(Long id) {  
        this.id = id;  
    }  
}
```

//čak i ukoliko ne stavimo anotaciju polja su prevedena u istoimene kolone.

//u prevođenju sa XML konfiguracionim fajlom to nije slučaj

@Column(name="STUDENT_NAME")

```
public String getName() {  
    return name;  
}  
public void setName(String name) {  
    this.name = name;  
}...
```

Hibernate

- **Primer upotrebe Hibernate-a**

@Entity

@Table(name="Student")

```
public class Student {
```

```
    ...
```

```
    //CascadeType definiše da li će biti snimljena i Adresa prilikom snimanja Studenta
```

```
    @OneToOne(cascade={CascadeType.ALL})
```

```
    @JoinColumn(name="ADDRESS_ID")
```

```
    public AddressEntity getAddressEntity() {
```

```
        return address;
```

```
    }
```

```
    ...
```

```
}
```


Sadržaj

- Protokoli
- ODBC
- JDBC
- OLE DB
- ADO .NET
- Objektno-relaciono prevođenje
- Hibernate
- Entity Framework

Entity Framework

- **Entity Framework**
 - *ADO.NET Entity Framework*
 - besplatno ORM okruženje za .Net platformu
 - omogućava razrešenje problema korišćenja relacionih pojmova u objektno orijentisanim aplikacijama
 - u aplikacijama se koriste objekti umesto direktnog pristupa bazi podataka
 - poseduje svoj upitni jezik *Entity SQL*
 - napravljen za konceptualni nivo
 - apstrahuje particionisane podatke

Entity Framework

- **Entity Framework**
 - koristi konceptualni model podataka
 - *eng. Entity Data Model (EDM)*
 - model na strani aplikacije
 - klijentski model
 - interakcija se vrši sa modelom
 - ne sa bazom podataka
 - operacije se prevode i izvršavanju nad bazom podataka
 - sastoji se od entiteta
 - imaju osobine
 - nemaju ponašanje
 - sadrže veze sa drugim entitetima

Entity Framework

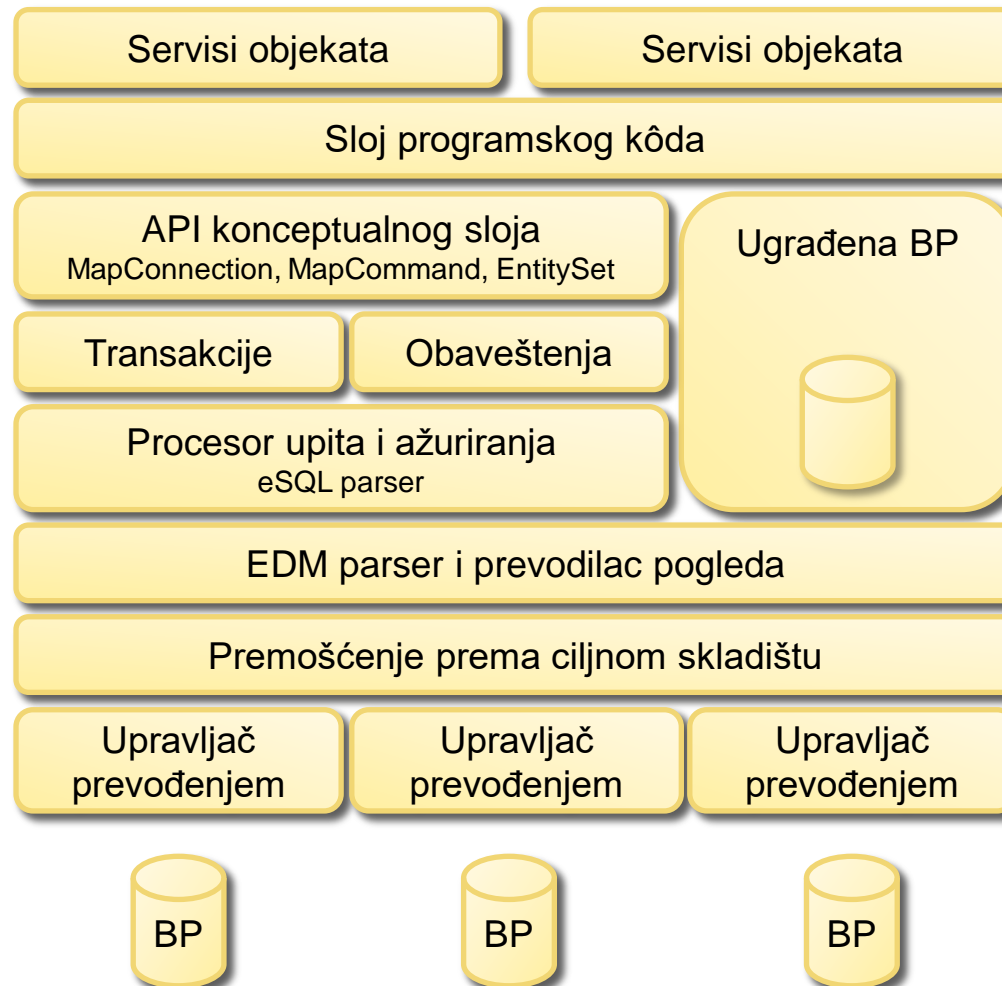
- **Entity Framework**

- EDM

- ne sadrži znanje o skladištu podataka
 - koristi upravljač podacima za komunikaciju sa BP
 - automatski generiše klase iz modela
 - omogućava standardno kreiranje upita
 - putem metoda
 - omogućava praćenje promena na modelima

Entity Framework

- Arhitektura Entity Framework-a



Entity Framework

- **Arhitektura Entity Framework-a**
 - **upravljač podacima**
 - specifični za svaki izvor podataka
 - apstrakcija ADO.NET interfejsa
 - za programiranje na konceptualnom nivou
 - **upravljač prevođenjem**
 - prevodi Entity SQL na SQL specifičan za ciljanu bazu podataka
 - uključuje **premošćenje prema ciljnom skladištu**
 - prevodi generičke naredbe u naredbe za neko skladište podataka

Entity Framework

- **Arhitektura Entity Framework-a**
 - **EDM parser i prevodilac pogleda**
 - parsira EDM i prevodi entitete na elemente iz relacione baze podataka
 - kreira poglede nad podacima u relacionoj BP
 - koji odgovaraju konceptualnom modelu
 - agregira podatke iz više tabela kako bi u jednom pogledu dobio informacije o entitetima
 - **procesor upita i ažuriranja**
 - obrađuje upite, filtere i ažuriranja
 - **usluge meta-podataka**
 - obrađuju sve meta-podatke vezane za entitete, veze i prevođenja

Entity Framework

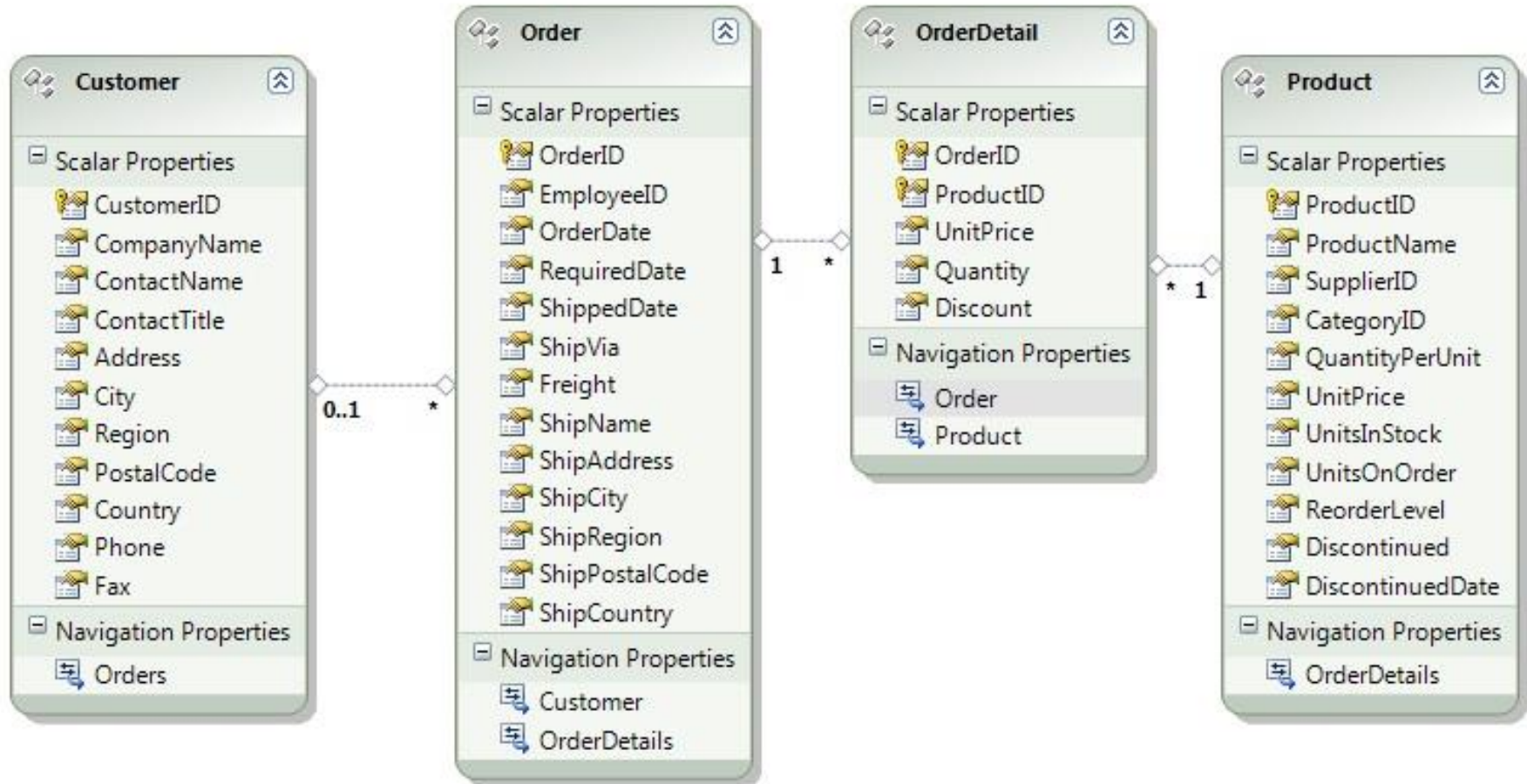
- **Arhitektura Entity Framework-a**
 - **transakcije**
 - integrisanje sa transakcijama BP
 - ukoliko BP ne podržava transakcije one se implementiraju u ovom sloju
 - **API konceptualnog sloja**
 - pruža interfejs ka modelu
 - koristi se iz kôda aplikacije
 - **odvojene komponente**
 - predstavljene ugrađenom BP
 - pružaju lokalni *cache*
 - **alati za dizajniranje**
 - lakše dizajniranje EDM-a
 - lakše prevođenje između entiteta i BP

Entity Framework

- **Arhitektura Entity Framework-a**
 - **sloj programskog kôda**
 - omogućava korišćenje EDM-a u konstruktima programskog jezika
 - dve vrste servisa
 - **servisi objekata**
 - » omogućavaju automatsko kreiranje klasa
 - » instanciranje entiteta kao objekata
 - **web servisi**
 - » omogućavaju kreiranje web servisa na osnovu EDM-a
 - **servisi visokog nivoa**
 - *reporting* servisi

Entity Framework

- Primer EDM-a



Entity Framework

- **Primer upotrebe Entity Framework-a**

```
public void EntityProviderPrimer1()
{
    // kreiranje veze koristeći upravljač podacima
    TdConnectionStringBuilder tbuilder = new TdConnectionStringBuilder();
    tbuilder.Database = "Database";
    tbuilder.DataSource = "Source";
    tbuilder.UserId = "user";
    tbuilder.Password = "password";

    EntityConnectionStringBuilder ebuilder = new EntityConnectionStringBuilder();

    // metapodaci su u EDM
    ebuilder.Metadata = @"res://Example/Model1.csdl|res://Example/Model1.msl" +
        "|res://Example/Model1.ssdl";

    ...
}
```

Entity Framework

- **Primer upotrebe Entity Framework-a**

```
public void EntityProviderPrimer1()
{
    ebuilder.Provider = "Teradata.Client.Provider";
    ebuilder.ProviderConnectionString = tbuilder.ToString();

    EDMExample context = new EDMExample(ebuild.ToString());

    //kreiranje upita koristeći EntitySql
    String queryStr = "SELECT o.OrderDate, od.ProductId, od.UnitPrice
                      FROM Orders AS o JOIN OrderDetails AS od ON
                      o.OrderID == od.OrderID";
    ObjectQuery<DbDataRecord> query =
        context.CreateQuery<DbDataRecord>(queryStr);

    ...
}
```

Entity Framework

- **Primer upotrebe Entity Framework-a**

```
public void EntityProviderPrimer1()
{
    ...
    // izvršavanje EntitySQL upita i preuzimanje osobina entiteta kao povratnih
    // vrednosti
    foreach(var result in query)
    {
        // NOTE: Da bi se vratio DATE, GetDateTime mora biti pozvan
        Console.WriteLine("OrderDate = {0}, ProductID = {1}, UnitPrice = {2}",
            result.GetDateTime(0), result.GetInt32(1), result.GetDecimal(2),
            result.UnitPrice);
    }
}
```

Reference

- **ODBC**
 - [http://msdn.microsoft.com/en-us/library/ms710238\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/ms710238(v=vs.85).aspx)
- **JDBC**
 - http://aspen.ucs.indiana.edu/webtech/jdbc/overviewpaper/JDBC_conn.html
- **OLE DB**
 - <http://msdn.microsoft.com/en-us/library/ms810892.aspx>
- **ADO .NET**
 - <http://msdn.microsoft.com/en-us/library/27y4ybxw.aspx>
- **ORM**
 - <http://www.agiledata.org/essays/mappingObjects.html>
- **Hibernate**
 - <http://www.site.lalitbhatt.com/hibernate-introduction>
- **Entity Framework**
 - <http://msdn.microsoft.com/en-us/library/bb399572.aspx>

Pitanja i komentari



Sadržaj

- Protokoli
- ODBC
- JDBC
- OLE DB
- ADO .NET
- Objektno-relaciono prevođenje
- Hibernate
- Entity Framework

Kraj prezentacije

Sistemi baza podataka



Obezbeđenje pristupa sistemu baze podataka

*Protokoli, ODBC/JDBC, ADO.NET,
OLE DB, O/R prevođenje*