Vývoj informačních systémů

Vzory: Objektově-relační chování

2021-22

Vztahy mezi vzory

- Vzory se nikdy nevyskytují osamoceně.
- Propojování vzorů je obvyklé zejména při spolupráci různých vrstev (logik).
- Každý katalog vzorů souvislosti popisuje.

Datové zdroje

- Table data gateway
  - An object that acts as a gateway to a database table. One instance handles all the rows in the table.
- Row data gateway
  - An object that acts as a gateway to a single record in a data source. There is one instance per row.
- Active record
  - An object that wraps a row in a database table or view, encapsulates the database access, and adds domain logic on that data.
- Data mapper
  - A layer of mappers that moves data between objects and a database while keeping them independent of each other and the mapper itself.

Table data gateway (kdy?)

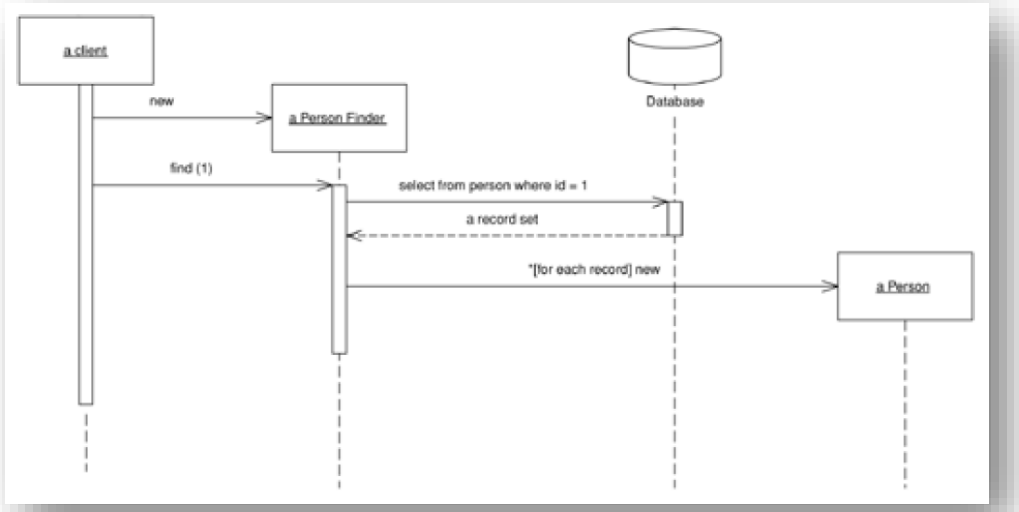
- + Jednoduchá doménová logika
- + S *Transaction Script* a *Table Module*
- - Ne s *Domain Model*
- + Záměna SQL logiky

```
class PersonGateway...

      public IDataReader FindAll() {
         String sql = "select * from person";
         return new OleDbCommand(sql, DB.Connection).ExecuteReader();
      }
      public IDataReader FindWithLastName(String lastName) {
         String sql = "SELECT * FROM person WHERE lastname = ?";
         IDbCommand comm = new OleDbCommand(sql, DB.Connection);
         comm.Parameters.Add(new OleDbParameter("lastname", lastName));
         return comm.ExecuteReader();
      }
      public IDataReader FindWhere(String whereClause) {
         String sql = String.Format("select * from person where {0}", whereClause);
         return new OleDbCommand(sql, DB.Connection).ExecuteReader();
      }
```

```
class PersonGateway...

      public Object[] FindRow (long key) {
         String sql = "SELECT * FROM person WHERE id = ?";
         IDbCommand comm = new OleDbCommand(sql, DB.Connection);
         comm.Parameters.Add(new OleDbParameter("key",key));
         IDataReader reader = comm.ExecuteReader();
         reader.Read();
         Object [] result = new Object[reader.FieldCount];
         reader.GetValues(result);
         reader.Close();
         return result;
      }
```

Row data gateway (kdy?)

- \+ S *Transaction Script*
- Jednoduchá doménová logika
- \- Ne s *Domain Model*




```
class PersonGateway...

   private static final String updateStatementString =
         "UPDATE people " +
         "  set lastname = ?, firstname = ?, number_of_dependents = ? " +
         "  where id = ?";
   public void update() {
      PreparedStatement updateStatement = null;
      try {
         updateStatement = DB.prepare(updateStatementString);
         updateStatement.setString(1, lastName);
```

Active Record (kdy?)

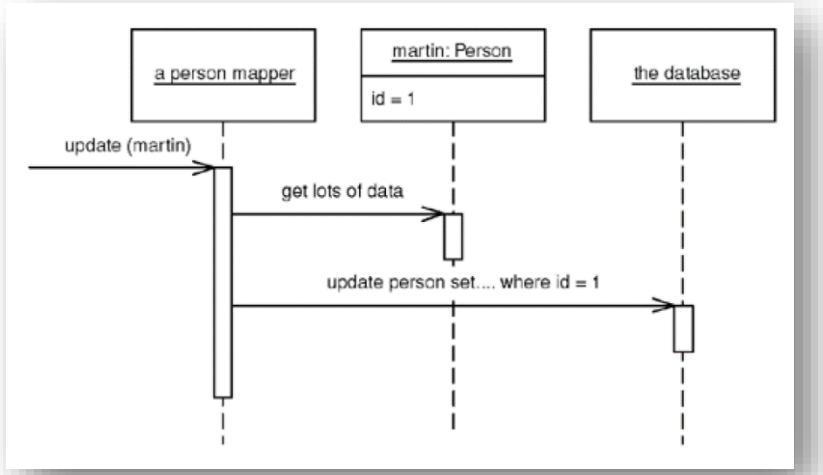
- \+ Složitější doména, ale s jednoduchými operacemi přímo mapovanými na tabulky.
- S *Table Data Gateway*
- \- Netriviální mapování do DB

```
class Person...

   private final static String findStatementString =
         "SELECT id, lastname, firstname, number_of_dependents" +
         "  FROM people" +
         "  WHERE id = ?";
   public static Person find(Long id) {
      Person result = (Person) Registry.getPerson(id);
      if (result != null) return result;
      PreparedStatement findStatement = null;
      ResultSet rs = null;
      try {
         findStatement = DB.prepare(findStatementString);
         findStatement.setLong(1, id.longValue());
         rs = findStatement.executeQuery();
         rs.next();
         result = load(rs);
         return result;
      } catch (SQLException e) {
         throw new ApplicationException(e);
      } finally {
         DB.cleanUp(findStatement, rs);
      }
   }
   public static Person find(long id) {
      return find(new Long(id));
   }
   public static Person load(ResultSet rs) throws SQLException {
      Long id = new Long(rs.getLong(1));
      Person result = (Person) Registry.getPerson(id);
      if (result != null) return result;
      String lastNameArg = rs.getString(2);
      String firstNameArg = rs.getString(3);
      int numDependentsArg = rs.getInt(4);
      result = new Person(id, lastNameArg, firstNameArg, numDependentsArg);
      Registry.addPerson(result);
      return result;
   }
```

```
public Money getExemption() {
   Money baseExemption = Money.dollars(1500);
   Money dependentExemption = Money.dollars(750);
    return baseExemption.add(dependentExemption.multiply(this.getNumberOfDependents(
}
```

Data Mapper (kdy?)

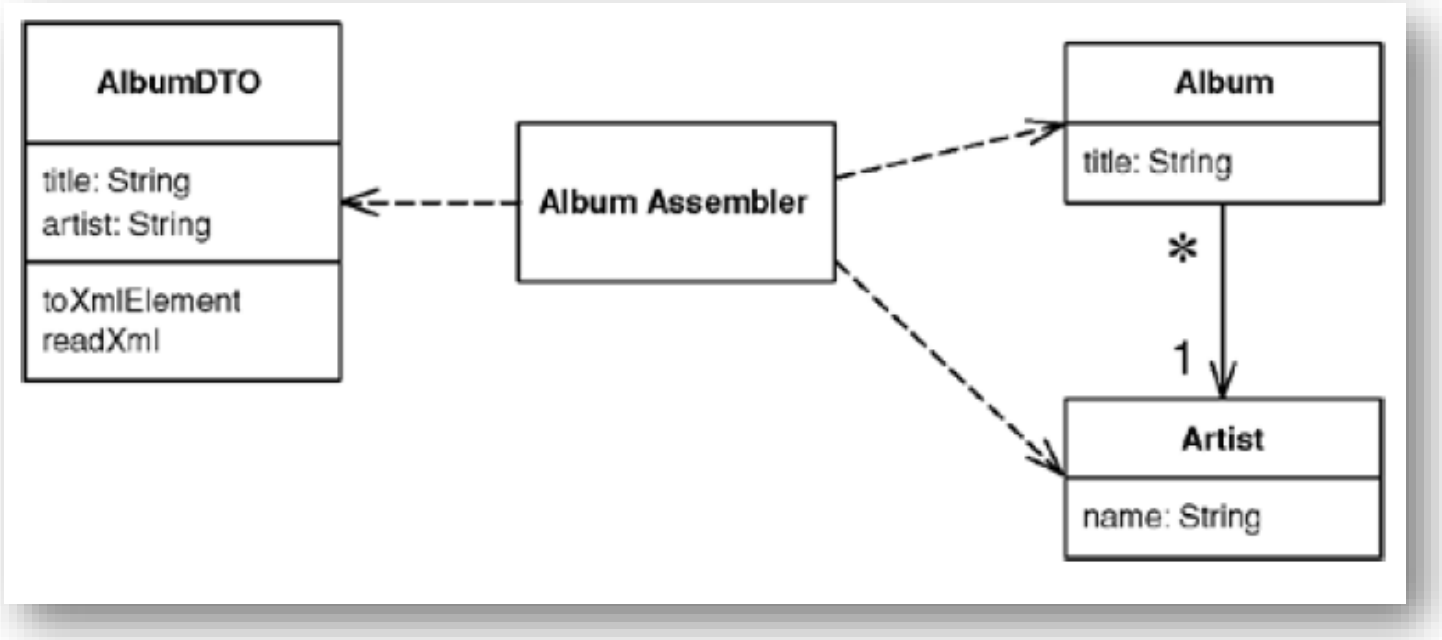
- + Nezávislá podoba domény a databáze
- + S *Domain Model*
- +Složitá doménová logika
- - Je komplikovanější




```
class PersonMapper...

   protected String findStatement() {
      return "SELECT " + COLUMNS +
         "  FROM people" +
         "  WHERE id = ?";
   }
   public static final String COLUMNS = " id, lastname, firstname, number_of_dependent
   public Person find(Long id) {
      return (Person) abstractFind(id);
   }
   public Person find(long id) {
      return find(new Long(id));
   }
```

Data Transfer Object (DTO)

- An object that carries data between processes in order to reduce the number of method.



Data, data, data…

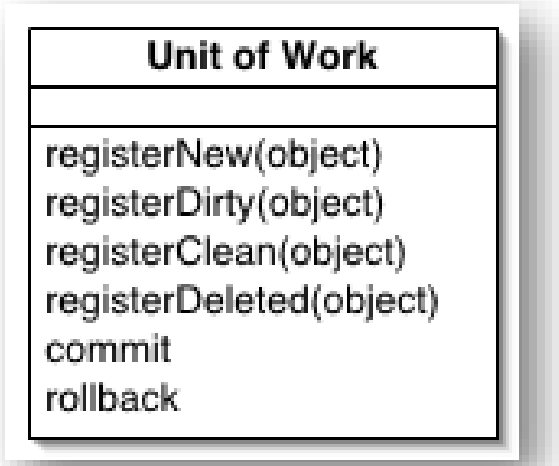
- Potřebujeme persistenci (programy se mění, data zůstávají).
- Mnoho dat (paměť nestačí).
- Přístup k datům (mnoho uživatelů a z různých míst, konkurence).

Objektově-relační chování

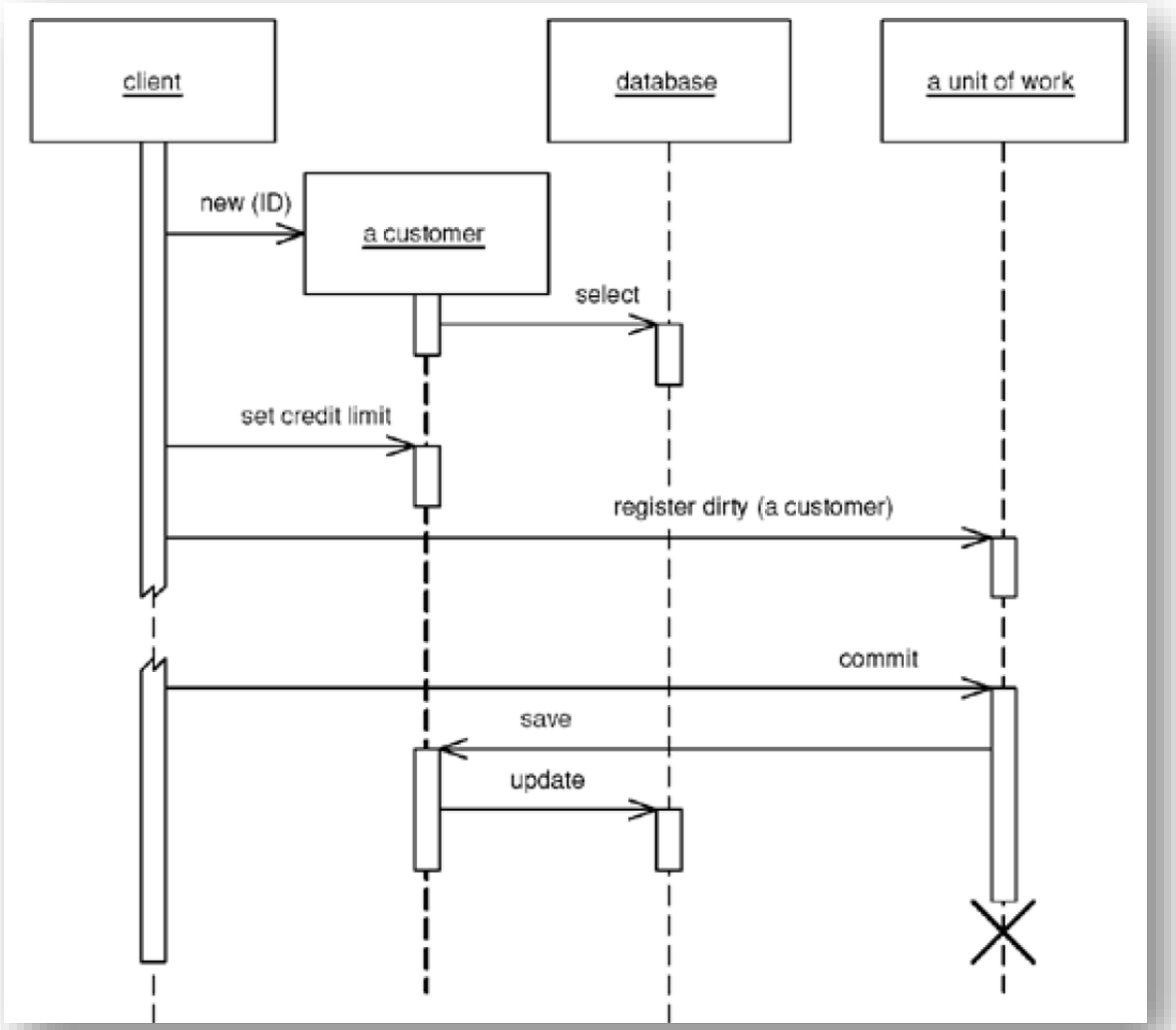
- Unit of Work
- Identity Map
- Lazy Load

Unit of Work

- Maintains a list of objects affected by a business transaction and coordinates the writing out of changes and the resolution of concurrency problems.

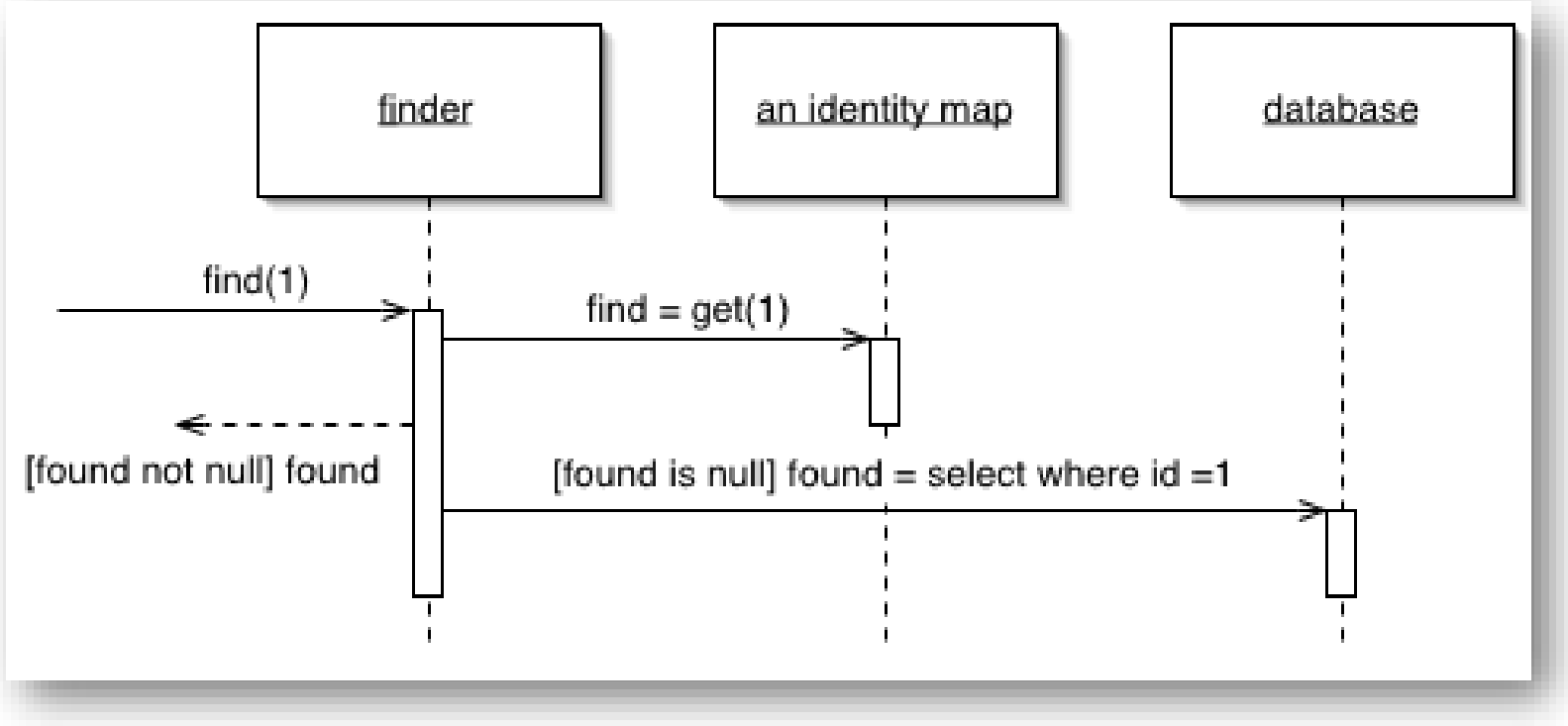


Průběh


client
database
a unit of work
new (ID)
a customer
select
set credit limit
register dirty (a customer)
commit
save
update

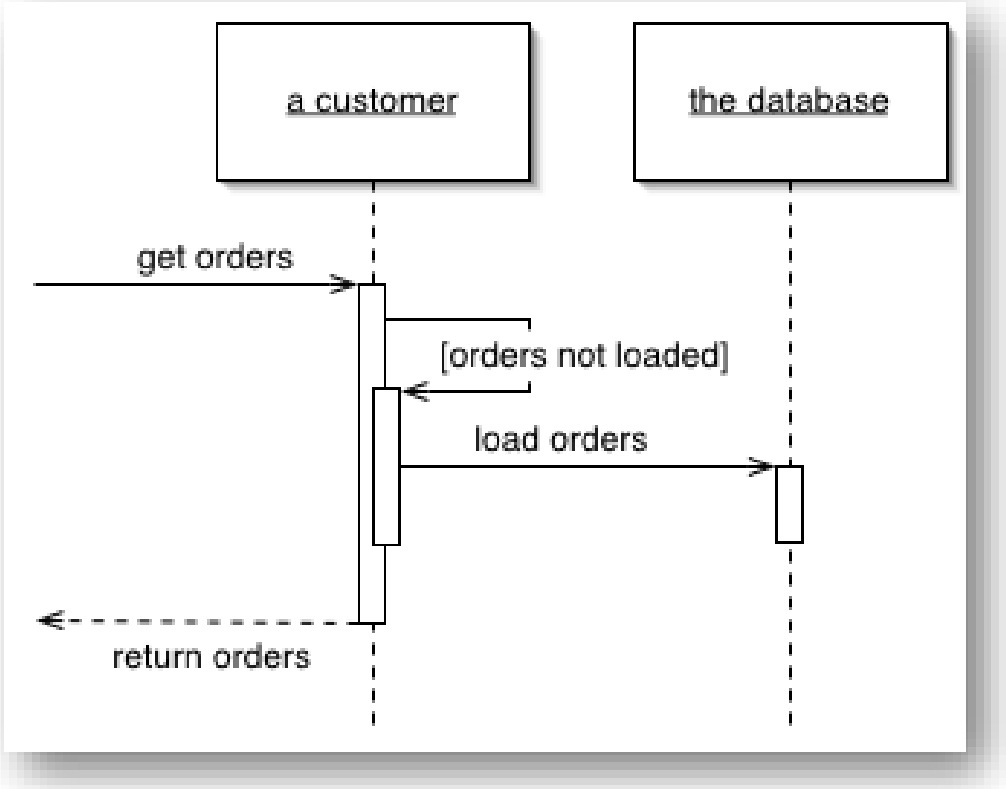
Identity Map

- Ensures that each object gets loaded only once by keeping every loaded object in a map. Looks up objects using the map when referring to them.



Lazy Load

- An object that doesn't contain all of the data you need but knows how to get it.



UI: Skica (wireframe, prototyp)

- Návrh definující funkci a obsah formulářů.
- Rozmístění funkčních prvků na stránce.
- Návrh navigace (odkud – kam – jak).
- Jednoduše (bez finálního vzhledu).

Úkoly na cvičení

- Prezentace připravených technických specifikací.
- Diskuze o prezentační vrstvě (uživatelské rozhraní, navigace).
- Implementace vzorů objektově-relačního chování pro třídy semestrálního úkolu.

Kontrolní otázky

1. Pro každý ze čtyř návrhových vzorů pro práci s datovými zdroji si promyslete a napište kousíček zdrojového kódu, ze kterého bude poznat, o který vzor jde.

2. Co společně řeší návrhové vzory pro objektově-relační chování?

3. Kdy bychom měli zvážit použití vzoru Unit of Work a proč? Na příkladu vysvětlete, jak jej použít.

4. Kdy bychom měli zvážit použití vzoru Identity Map a proč? Na příkladu vysvětlete, jak jej použít.

5. Kdy bychom měli zvážit použití vzoru Lazy Load a proč? Na příkladu vysvětlete jak jej použít.

6. Popřemýšlejte, se kterými vzory pro práci s datovými zdroji byste mohli společně použít některé ze vzorů pro objektově-relační chování a proč? Zkuste najít příklady.

K přečtení…

- Martin Fowler. *Patterns of Enterprise Application Architecture.* Addison-Wesley Professional, 2003 [184-214].