Struktura a design JSP-servlet aplikací

Design aplikací

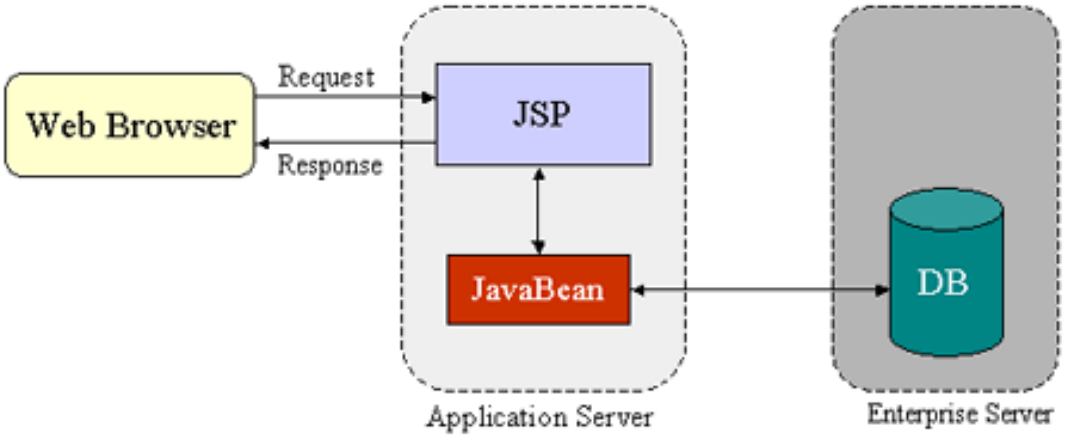
- Základ: MVC jako vždy
  - nemíchat logiku s prezentací
- Vrstvy a jejich realizace
  - prezentace: JSP
    - » v JSP žádný skriptovaný kód
  - logika, řízení: servlety (+ pomocné třídy, EJB, …)
  - data: JavaBeans, DTOs (+ EJB, ORM)
    - » nastaveny servletem, čteny JSP
    - » případně i obráceně

Architektury Java webových aplikací

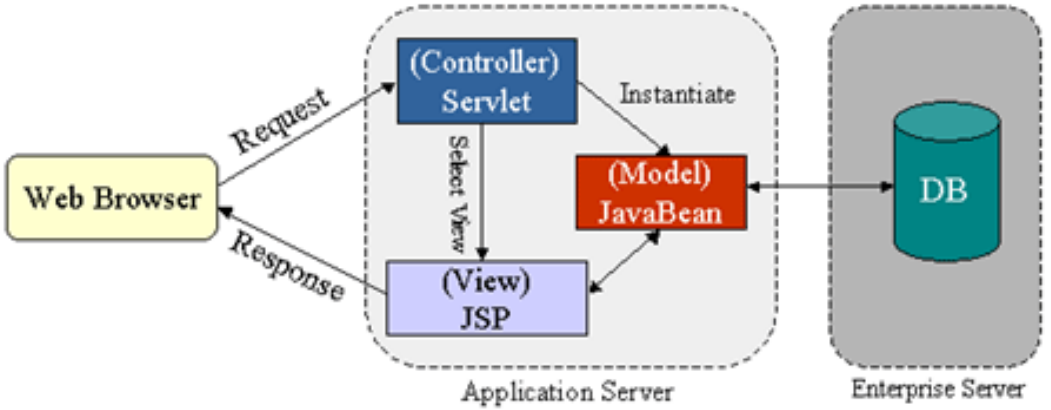
- Vždy vícevrstvá architektura
- Malé aplikace
  - web kontejner se servlety/JSP v aplikační vrstvě
  - servlety propojeny s databází
- Enterprise aplikace
  - EJB kontejner s aplikační logikou a přístupem k db
  - nebo aplikační vrstva + ORM
  - JSP, JSF nebo Struts pro webovou vrstvu

Modely integrace servletů a JSP

- „Model 1“
  - pouze triviální aplikace

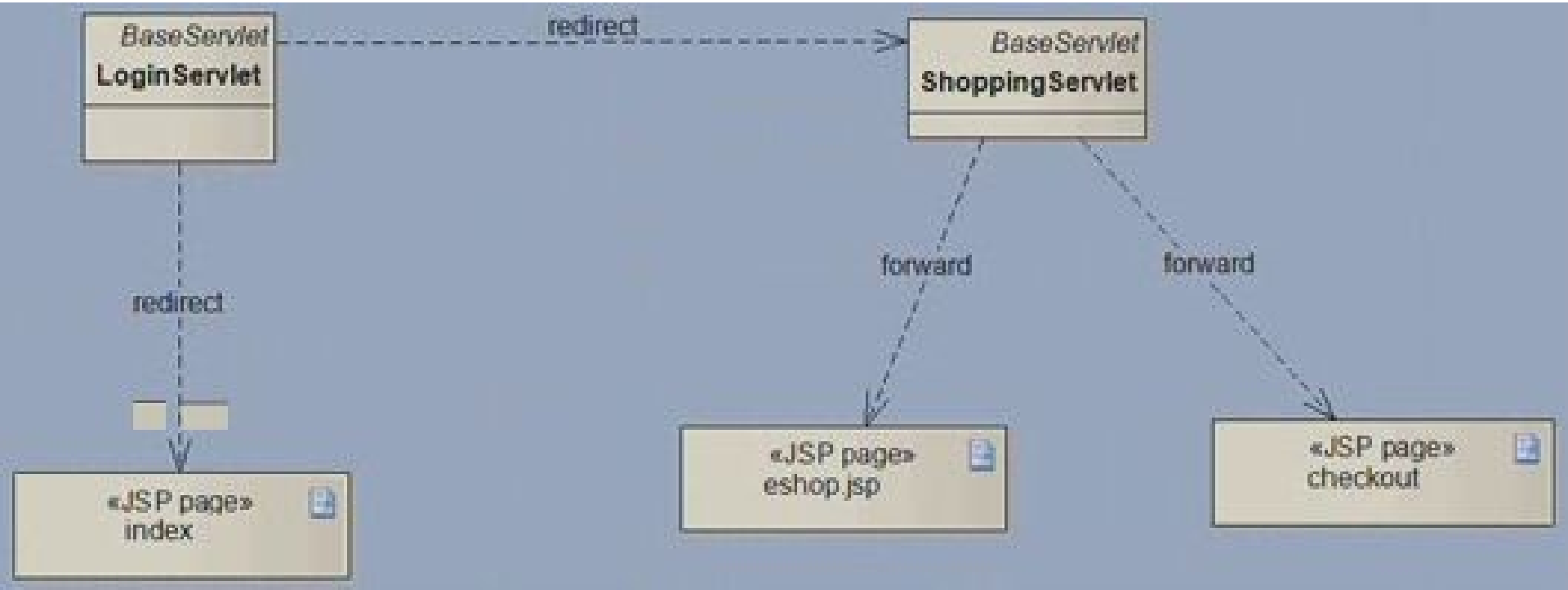


- „Model 2“
  - preferovaný
  - front controller + logika

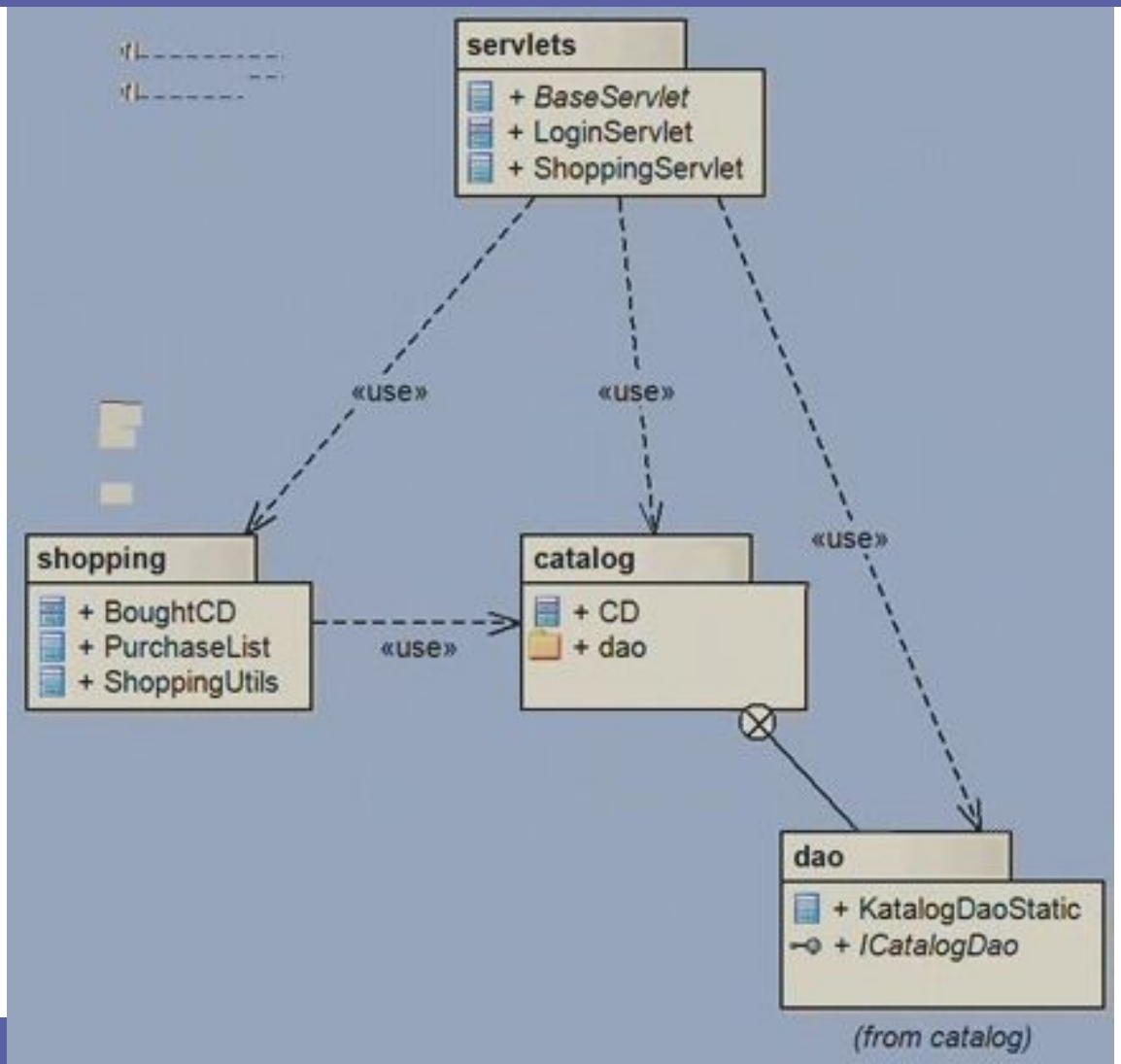


Příklad Model 2 struktury

- Lze stáhnout s portálu
  - odkaz Příklad MVC aplikace s JSTL z přednášky
    - v sekci Samostatná práce

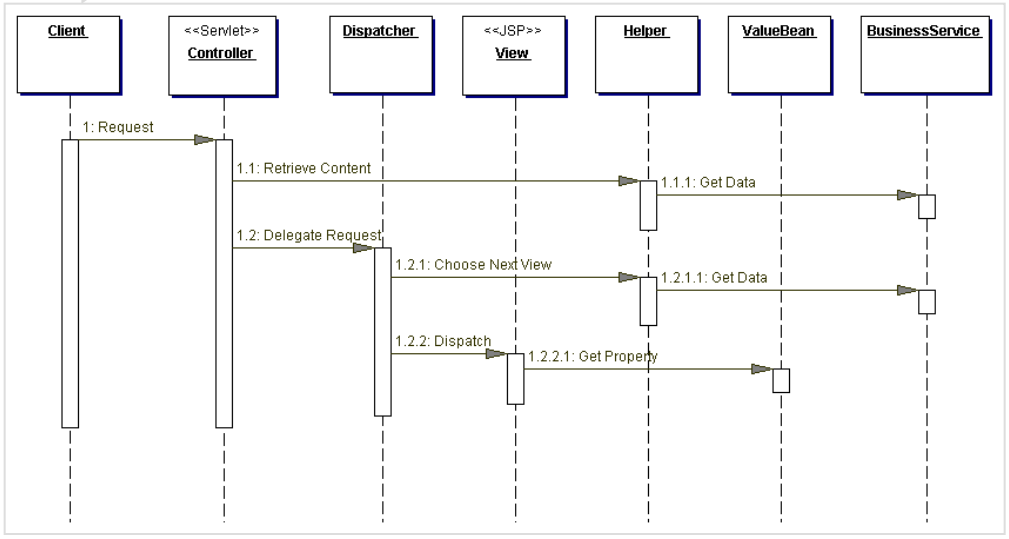


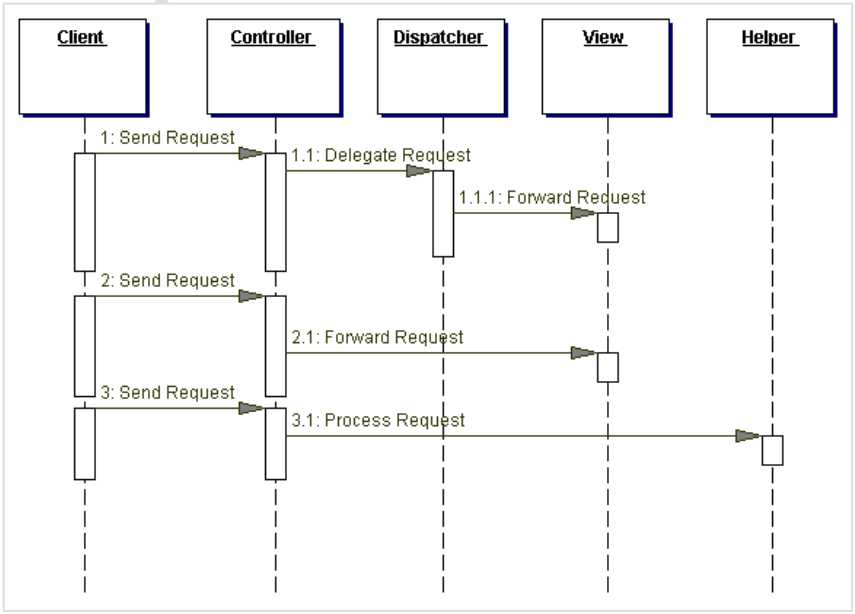
Příklad Model 2 struktury



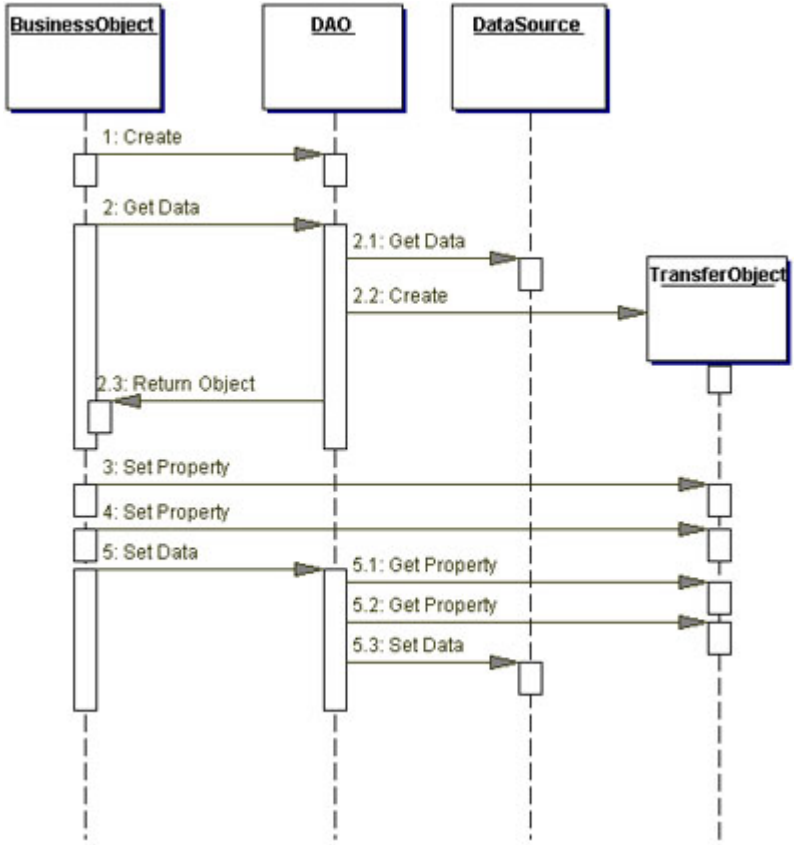
Návrhové vzory pro J2EE aplikace

- Front Controller
- Dispatcher View, Service to Worker
- Business Delegate (delegát)
- Data Access Object – DAO





Návrhové vzory – DAO




```
public interface StudentDAO {

   List<Student> selectStudents();

   void insertStudent(Student student);

   void updateStudent(Student student);

   void deleteStudent(Student student);

   List<Student> selectStudentsOnLecture
                          (Lecture lecture);

   List<Student> selectStudentsByName
               (String first, String last);

}
```

Další best practices

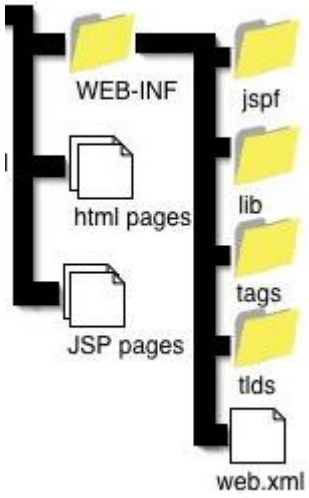
» Enterprise Java Blueprints

- Psaní JSP
  - používat EL – lepší čitelnost, snazší práce
  - custom tags místo scriptletů/tříd pro často používané funkce
  - reuse ověřených knihoven (JSTL, Jakarta Taglibs)
- Vkládání obsahu
  - statický, dynamický a oddělený obsah
  - @*include* × *<jsp:include …>* × *<c:import …>*
- Práce s daty
  - používat connection pooling
    - » vyžaduje práci s JNDI
  - ukládat (cache) výsledy dotazů – *ArrayList* místo *ResultSet*

Adresářová struktura aplikace

- Cíl: přehlednost, udržovatelnost, škálování

- JSP
  - samostatný adresář pro fragmenty
  - (advanced) oddělení kanálů
    - » HTML, WML, RSS, PDF, …
- Lokalizace
  - resource bundle třídy, JSP s texty



- Vývoj
  - oddělení Java src do samostatného stromu
  - ant build.xml v nadřazeném adresáři

Rámce pro 3-vrstvé aplikace

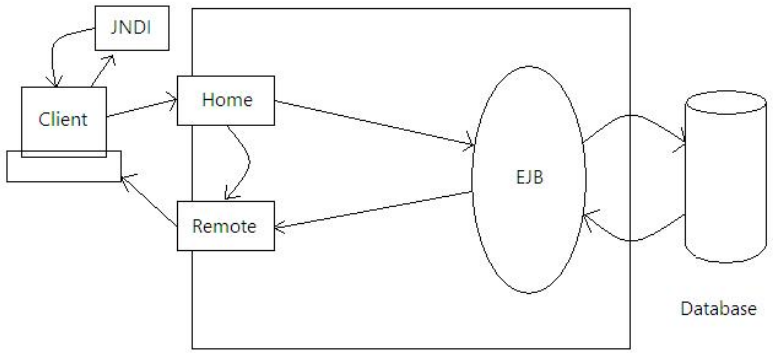
Webové frameworky

- Složité a velké aplikace → klasický Model 2 pracný

- Jakarta Struts (Apache)
  - kontrolér, akce + podpora view
  - http://struts.apache.org/
- JavaServer Faces (Sun)
  - view, řízení událostmi
  - kontrolér deklarativní
  - http://java.sun.com/j2ee/javaserverfaces/
- Spring MVC Framework
  - http://www.springframework.org/docs/reference/mvc.html

Technologie pro enterprise aplikace

- Enterprise JavaBeans (EJB)
  - komponentový model pro aplikační vrstvu, vzdálený přístup
    - home a remote interface, XML deployment desc, JNDI lookup
  - SessionBeans: aplikační logika
    EntityBeans: datová abstrakce
    MessageDrivenBeans: obsluha asynchronních událostí
  - kontejner (JBoss, WebLogic, JOnAS, …)
  - nejpoužívanější verze 2.1, od 3.0 konverguje k POJO a ORM



Technologie pro enterprise aplikace (2)

- Portlety a portály
  - framework pro tvorbu aplikací s webovým rozhraním
    - single sign-on
    - metafora desktopu
    - integrace back-end aplikací (nové i legacy)
      - » na prezentační úrovni
  - portlet = funkčně ucelená malá část aplikace
    - založeno na servletech a JSP
    - view, edit, config, help režimy
    - komunikace mezi portlety
  - portál = kontejner (Pluto, Jetspeed, WebSphere, …)
  - standardy API: IBM, JSR168

Technologie pro enterprise aplikace (3)

- Webové služby (web services)
  - implementace SOA – Service-Oriented Architecture
  - RPC/ROI přes HTTP
    - data (parametry, výsledky) v XML
  - SOAP implementace (Simple Object Access Protocol)
    - popis rozhraní v WSDL (XML) + UDDI
    - zpráva: obálka, tělo, chybové stavy
  - REST implementace (Representational State Transfer)
    - všechny 4 metody HTTP/1.1
    - lze použít URI pro jednoduché zprávy