Laboratorio di Basi di Dati/ Basi di dati per Bioinformatica

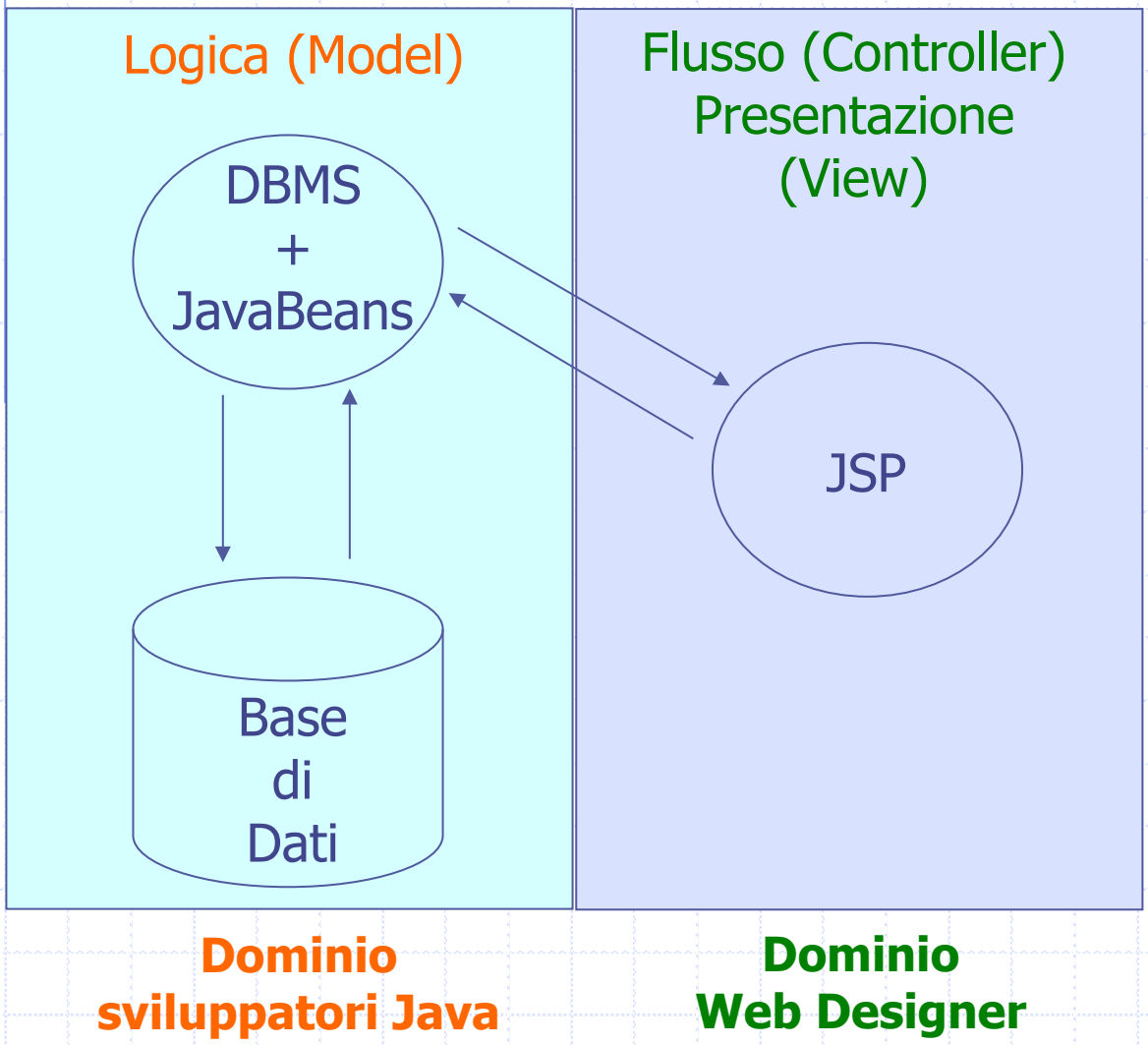
Docenti: Alberto Belussi e Carlo Combi

Lezione 9

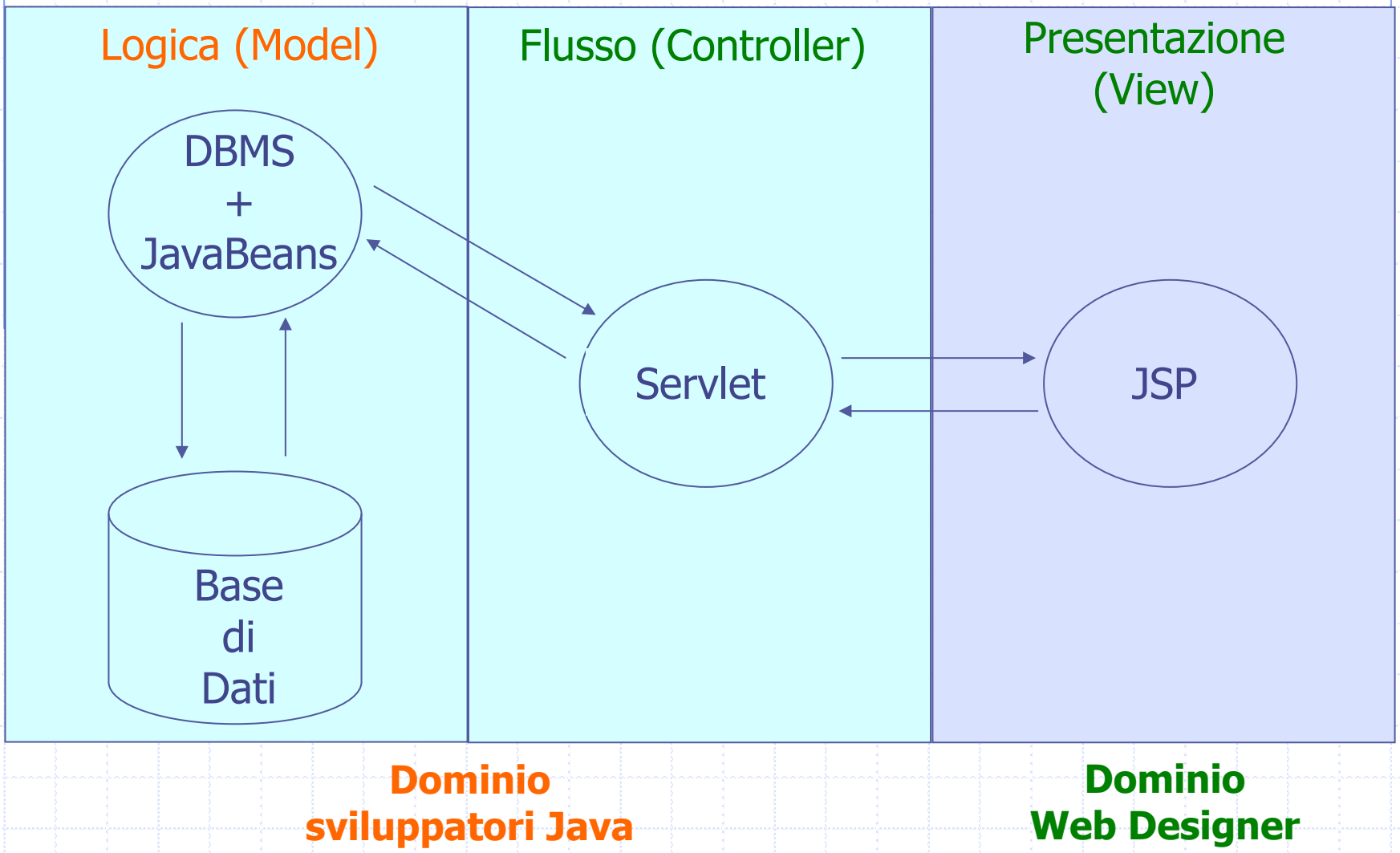
Architettura Model-View-Controller (MVC)

- Adottando l'architettura MVC e la tecnologia Servlet-JSP, un'applicazione web può essere realizzata secondo diversi approcci.
- I due approcci più significativi sono:
  - page-centric
  - servlet-centric

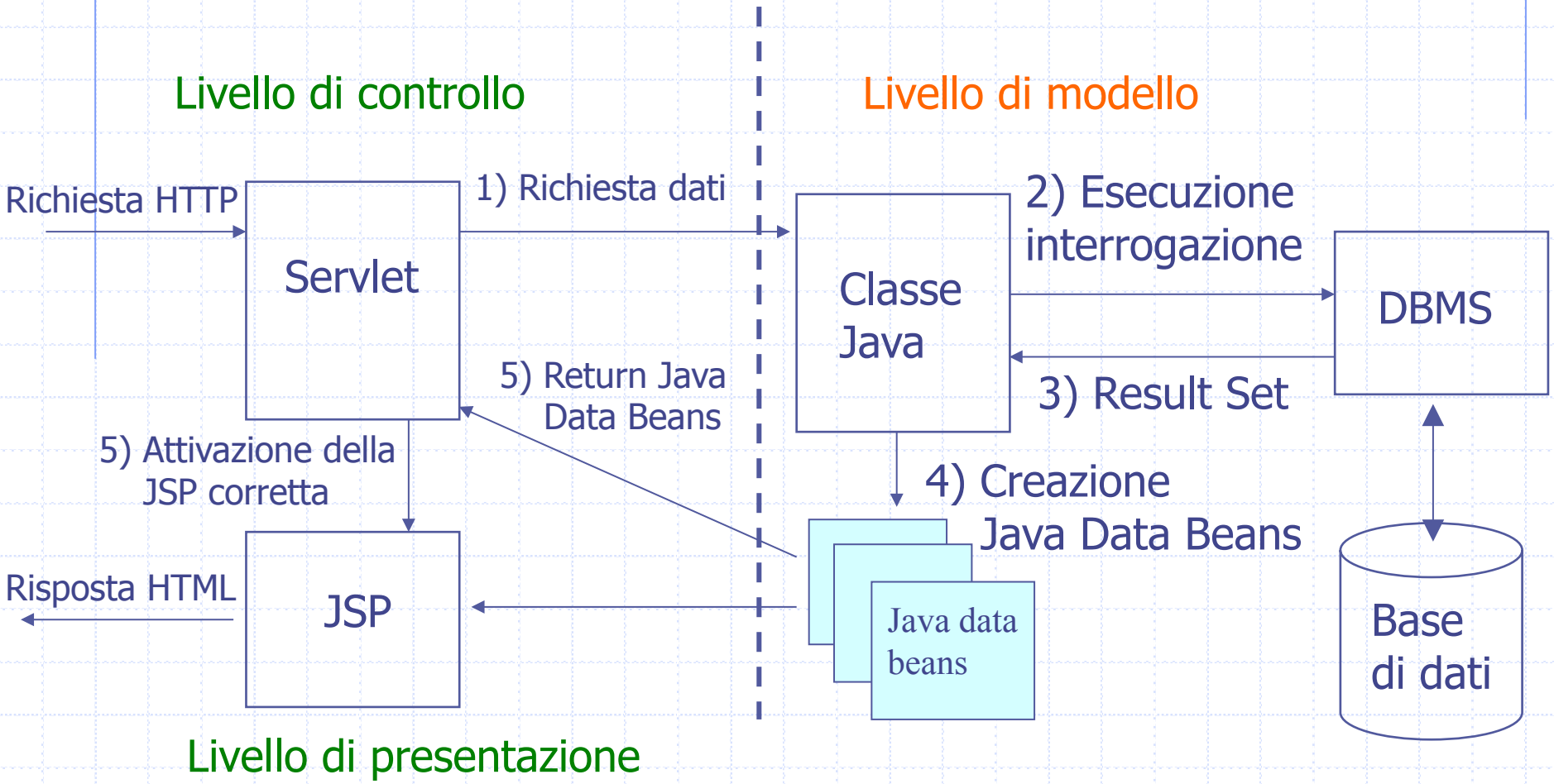
Approccio Page-centric



Approccio Servlet-centric (1)



Approccio Servlet-centric (2)



Approccio Servlet-centric (3)

- L'approccio servlet-centric prevede di utilizzare le pagine JSP solo per la presentazione e delegare il controllo ad una o più servlet.
  Le servlet quindi:
  - gestiscono le richieste (vengono cioè invocate tramite URI)
  - elaborano i dati necessari a soddisfare le richieste (interagendo con la classe DBMS.java e utilizzando i JavaDataBean come componenti per rappresentare le informazioni di interesse)
  - trasferiscono il controllo alla JSP designata a presentare i risultati.

Approccio Servlet-centric (4)

- **Passaggio dati fra servlet-JSP:**

  i JavaDataBean istanziati dalla servlet devono essere passati alla JSP prima di trasferire ad essa il controllo. A tal fine esiste una coppia di metodi della classe HttpServletRequest che permettono di inserire/recuperare in/da request (oggetto implicito della JSP) un numero arbitrario di oggetti. Questi metodi sono:

  - setAttribute(String, Object)
  - getAttribute(String)

Approccio Servlet-centric (5)

- **Trasferimento del controllo dalla servlet alla JSP:** quando all'interno di una servlet, dopo aver preparato i dati e averli inseriti nell'oggetto *request*, si vuole richiamare una JSP per visualizzare i dati, si dice che si *trasferisce il controllo (forward)* alla JSP.

Approccio Servlet-centric (6)

- Per trasferire il controllo è necessario creare un oggetto di tipo RequestDispatcher associato alla JSP che si vuole 'invocare'.
- Ci sono due modi equivalenti per definire un oggetto RequestDispatcher associato ad una JSP all'interno di una servlet:
  - RequestDispatcher rd = req.getRequestDispatcher("**PathRelativoJSP**")
  - RequestDispatcher rd = getServletContext.getRequestDispatcher ("**PathAssolutoJSP**")

Approccio Servlet-centric (7)

- Una volta ottenuto l'oggetto RequestDispatcher, è sufficiente invocare il metodo

  forward(HttpServletRequest, HttpServletResponse)

  per trasferire MOMENTANEAMENTE il controllo alla JSP.

- **Attenzione!** Non è un browser redirect e nemmeno una terminazione del metodo doGet o doPost della servlet... è una semplice chiamata di metodo. Tutto il codice presente DOPO forward(HttpServletRequest, HttpServletResponse) verrà eseguito dopo che la JSP ha finito la sua esecuzione!

Dalla progettazione logica (page-schema) all'architettura MVC

E' possibile indicare alcuni criteri per la traduzione di un progetto logico di un sito web centrato sui dati in un insieme di moduli (servlet, JSP e JDB) dell'approccio MVC servlet-centric.

Dalla progettazione logica (page-schema) all'architettura MVC

Assegnato uno schema logico costituito da un insieme di schemi di pagina (SP1, …, SPn) e dalle interrogazioni SQL (Q1, …, Qm) che li alimentano:

- Si genera una servlet main.java per il controllo di flusso.
  La servlet main gestisce tutte le richieste HTTP e suppone presente un parametro "ps" nella richiesta HTTP, che indica quale schema di pagina viene richiesto.

Dalla progettazione logica (page-schema) all'architettura MVC

- Per ogni interrogazione Q_i si genera un JDB per rappresentare le tuple del risultato della sua esecuzione.
- Alcuni bean potrebbero essere molto simili tra loro, in tal caso unire tra loro i JDB simili per generare un unico JDB.
- Tutte le interrogazioni SQL vengono gestite dalla classe DBMS.java che include un metodo getXXX per ogni interrogazione Qi e un metodo makeYYYBean per ogni tipo di JDB che deve creare.

Dalla progettazione logica (page-schema) all'architettura MVC

- I parametri delle interrogazioni devono essere gestiti come parametri delle richieste HTTP e quindi recuperati dalla servlet main e passati ai metodi getXXX.
- Il contenuto informativo degli schemi di pagina deve essere prodotto dalle JSP. Si genera una JSP per ogni schema di pagina SPi.
- In tali JSP la generazione di link deve essere fatta ponendo attenzione anche ai parametri che vengono richiesti dallo specifico schema di pagina verso il quale si sta generando il link.

Esempi da scaricare

1. Scaricare nella directory ~/tomcat/webapps/CorsoStudi le JSP: ElencoCorsiStudio.jsp e daFare.jsp dalla pagina web del corso.
2. La JSP ElencoCorsiStudio.jsp consente la visualizzazione dei corsi di studio dell'ateneo. La JSP daFare.jsp segnala che la funzionalità è da implementare.
3. Per far funzionare questo esempio è necessario:
    1. Restando nella directory ~/tomcat/src/CorsoStudi scaricare il file main.java.
    2. Compilare il package did e la servlet main nel seguente modo: javac –d ../../../webapps/CorsoStudi/WEB-INF/classes main.java ./did/*.java
    3. Dichiarare la servlet main nel file web.xml in webapps/CorsoStudi/WEB-INF

Esempi da scaricare

4. Per vedere le pagine web prodotte dall'applicazione (come dichiarato nel file web.xml)
   : http://localhost:8080/CorsoStudi/servlet/main
5. Completare la conversione dell'applicazione CorsoStudi all'architettura MVC servlet-centric (vedi esercizio proposto).
   5. Modificare la servlet main per la gestione del flusso di esecuzione, con i parametri indicati nel testo dell'esercizio.
   6. Aggiungere le due JSP mancanti per la visualizzazione delle informazioni sul singolo corso di studi e della lista di insegnamenti per anno accademico.

Riferimenti

- Marty Hall. “CORE. Servlets and JavaServer Pages”. Sun Microsystems Press.
- Phil Hanna. “JSP. La guida Completa.” McGraw-Hill.