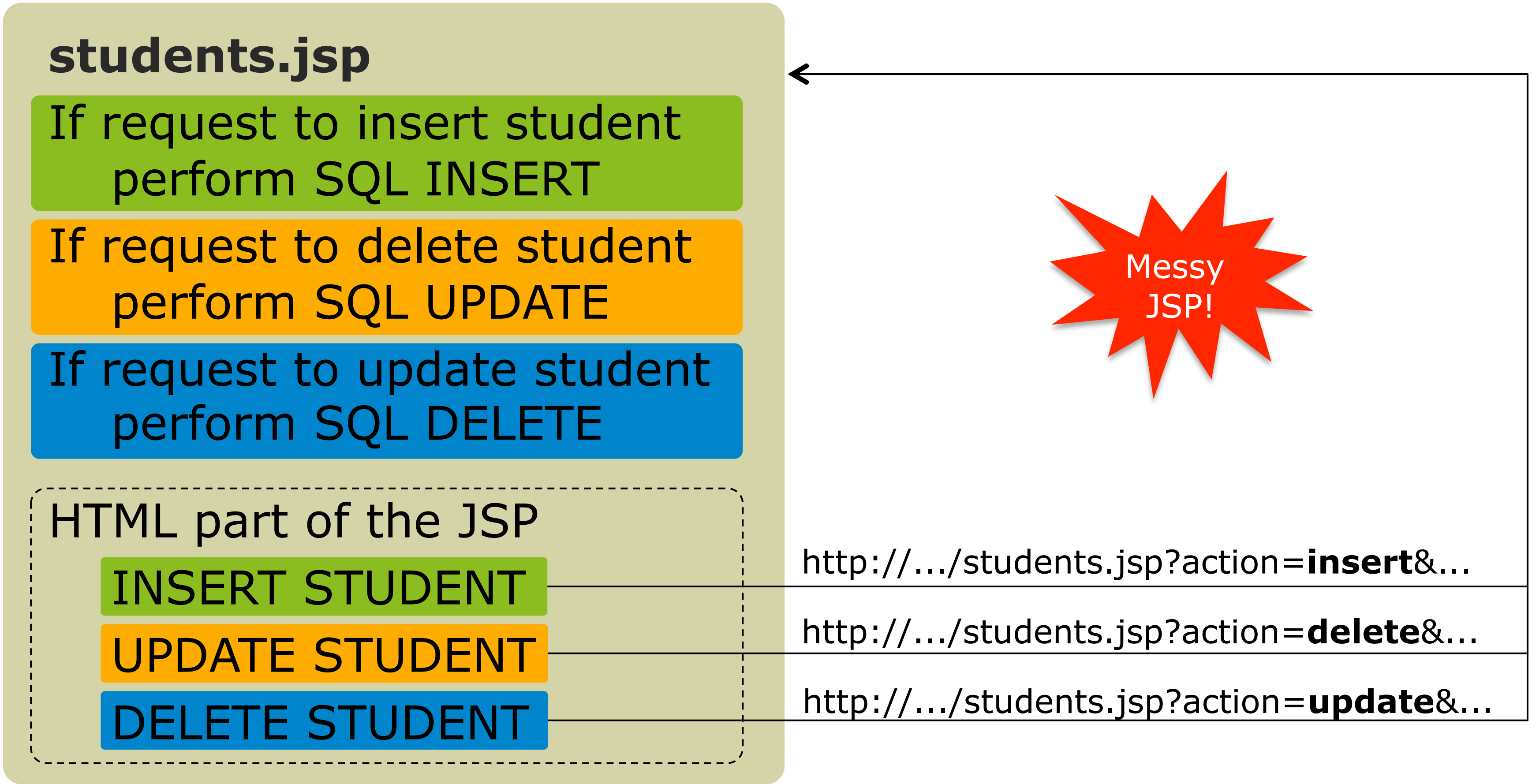
CSE 510
Web Data Engineering

The MVC Design Pattern &
The Struts Framework

cse@buffalo

Previous Attempts: Model 1 Design Pattern

```
for every JSP page p
    for every type of request r to p
        insert in p code to implement the action requested by r
```



students.jsp

If request to insert student
perform SQL INSERT

If request to delete student
perform SQL UPDATE

If request to update student
perform SQL DELETE

HTML part of the JSP

INSERT STUDENT

UPDATE STUDENT

DELETE STUDENT

http://.../students.jsp?action=**insert**&...

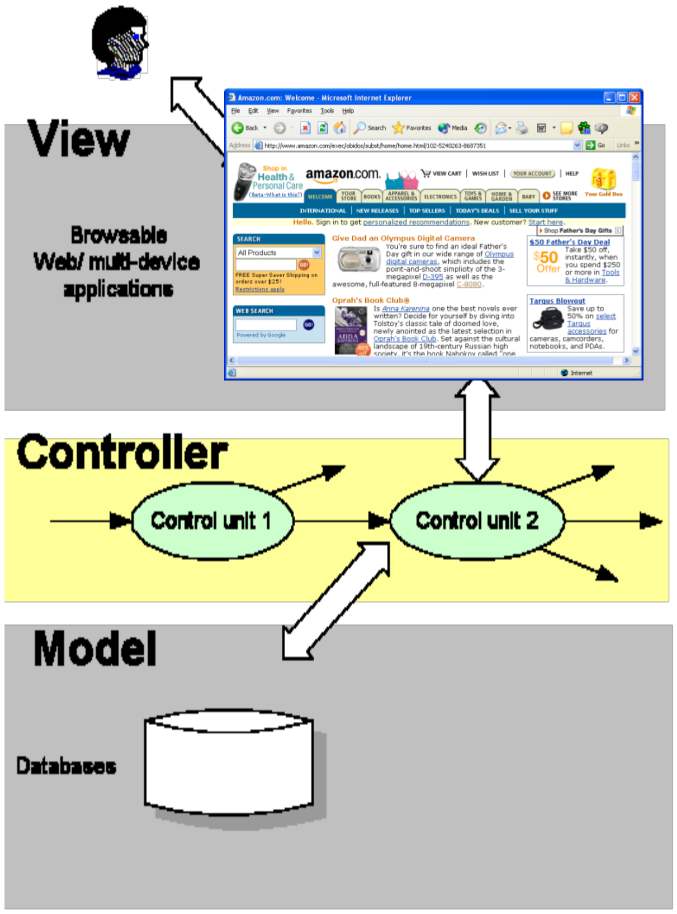
http://.../students.jsp?action=**delete**&...

http://.../students.jsp?action=**update**&...

Messy JSP!

The MVC Design Pattern: Separating Model, View & Controller

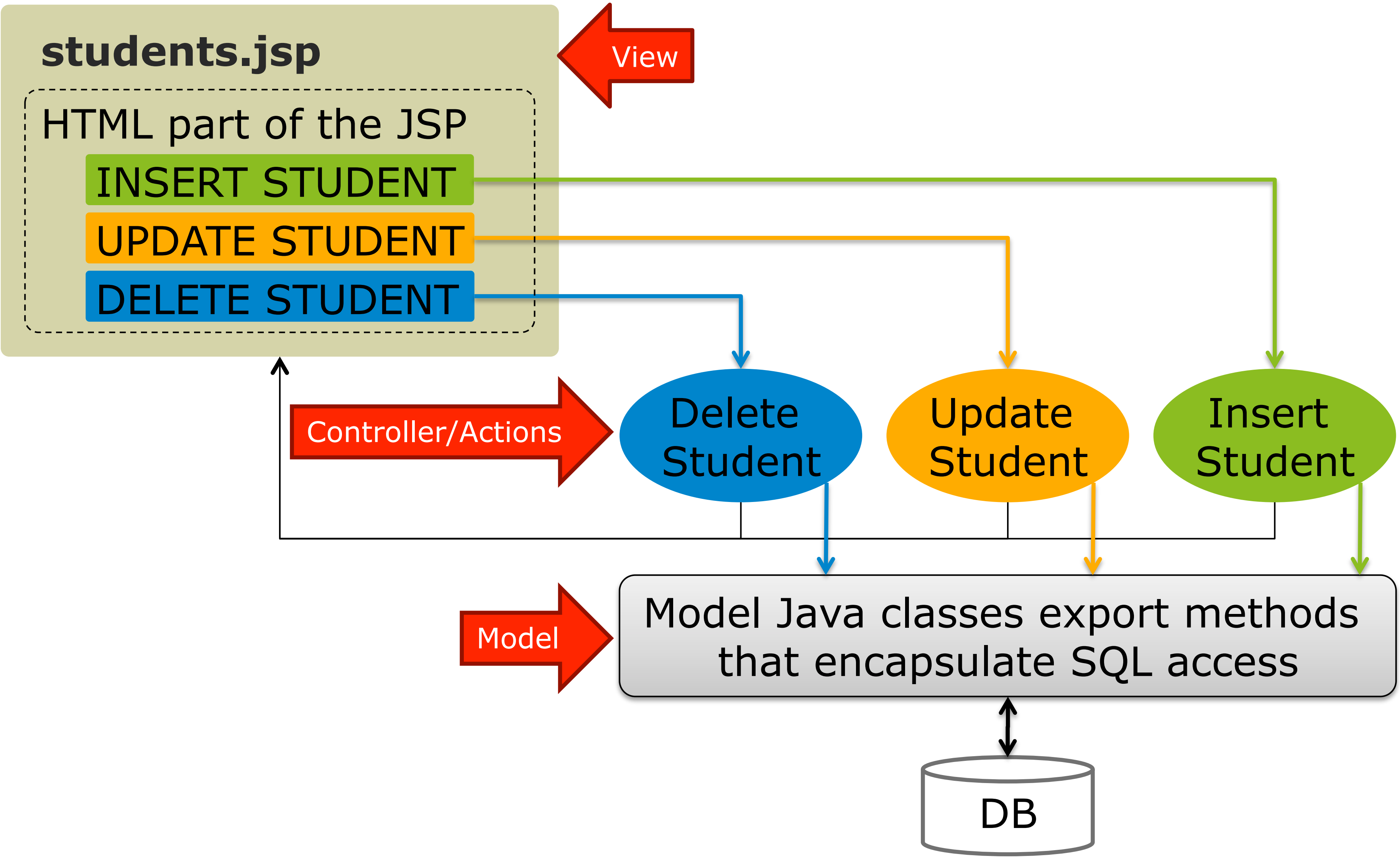
- Development "Best Practice"
- Known well before web items
  - Smalltalk pioneered
- **Model**: Access to Underlying Databases and Info Sources
- **Controller**: Control Flow of Web App
- **View**: Look-and-Feel



The MVC Design Pattern

- MVC originated as Model 2 in web developers community
- **Model 1**: Application logic is attached to JSPs
  - Similar to previous attempts of students.jsp
- **Model 2**: Data access and control flow decisions in Java Beans

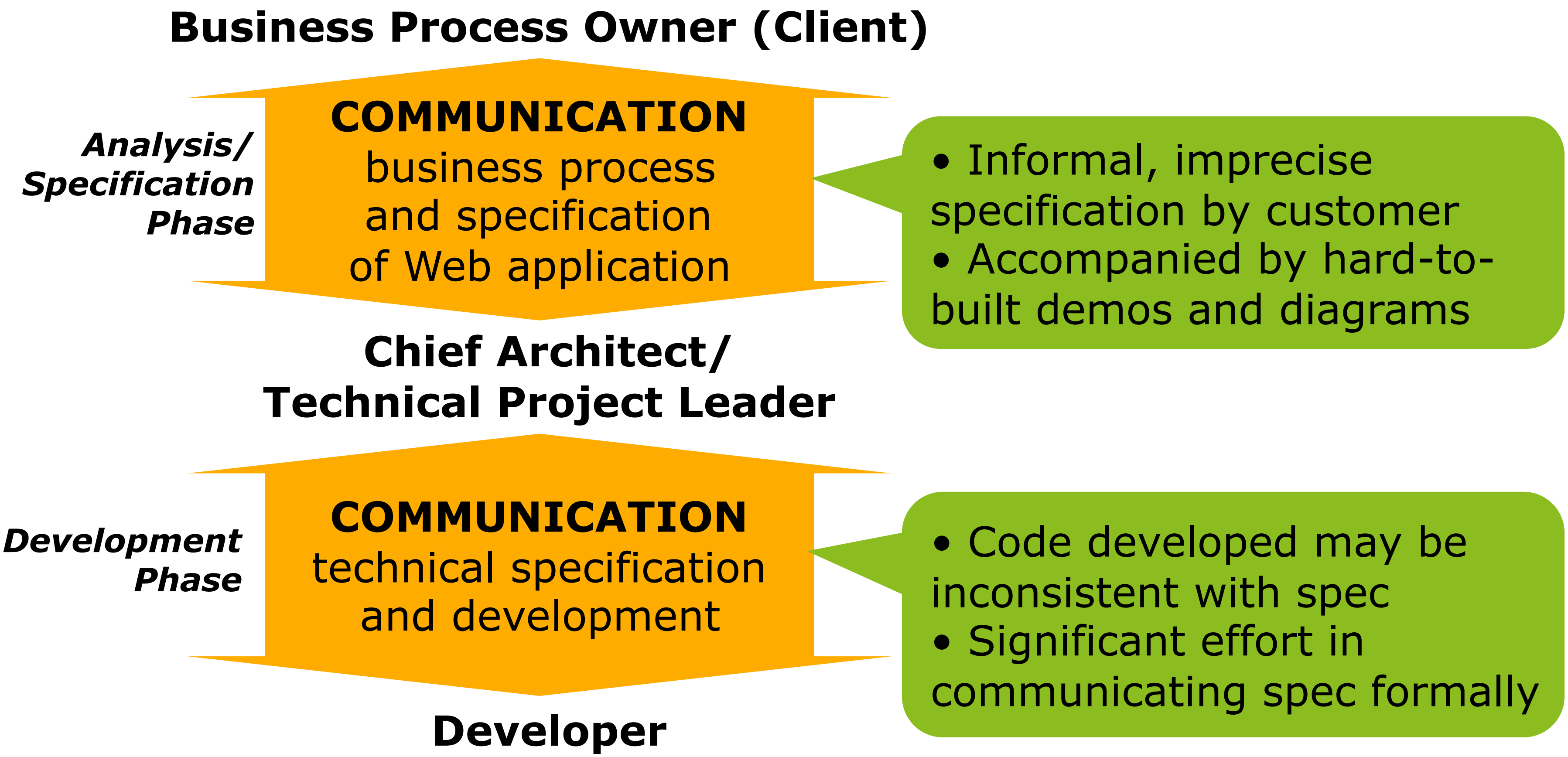
Data Entry Example – MVC Attempt



The Larger Issue: Specification and Modularization

- Frictions in Specification
- Inefficiencies in Large Project Management

The Process and the Frictions



Business Process Owner (Client)

Analysis/ Specification Phase

COMMUNICATION
business process
and specification
of Web application

- Informal, imprecise specification by customer
- Accompanied by hard-to-built demos and diagrams

Chief Architect/
Technical Project Leader

Development Phase

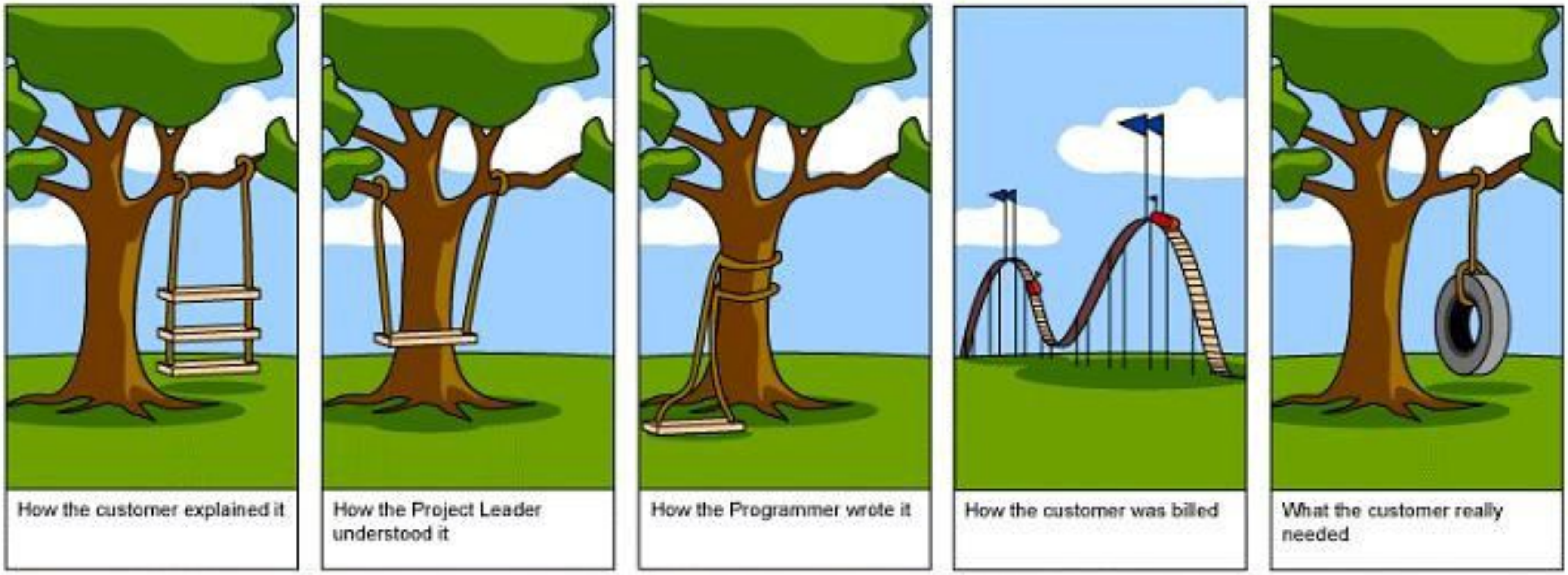
COMMUNICATION
technical specification
and development

- Code developed may be inconsistent with spec
- Significant effort in communicating spec formally

Developer

Problem is even worse in evolution phase when application logic is hidden in thousands of lines of code

The Problem: Communication


How the customer explained it
How the Project Leader understood it
How the Programmer wrote it
How the customer was billed
What the customer really needed

Struts

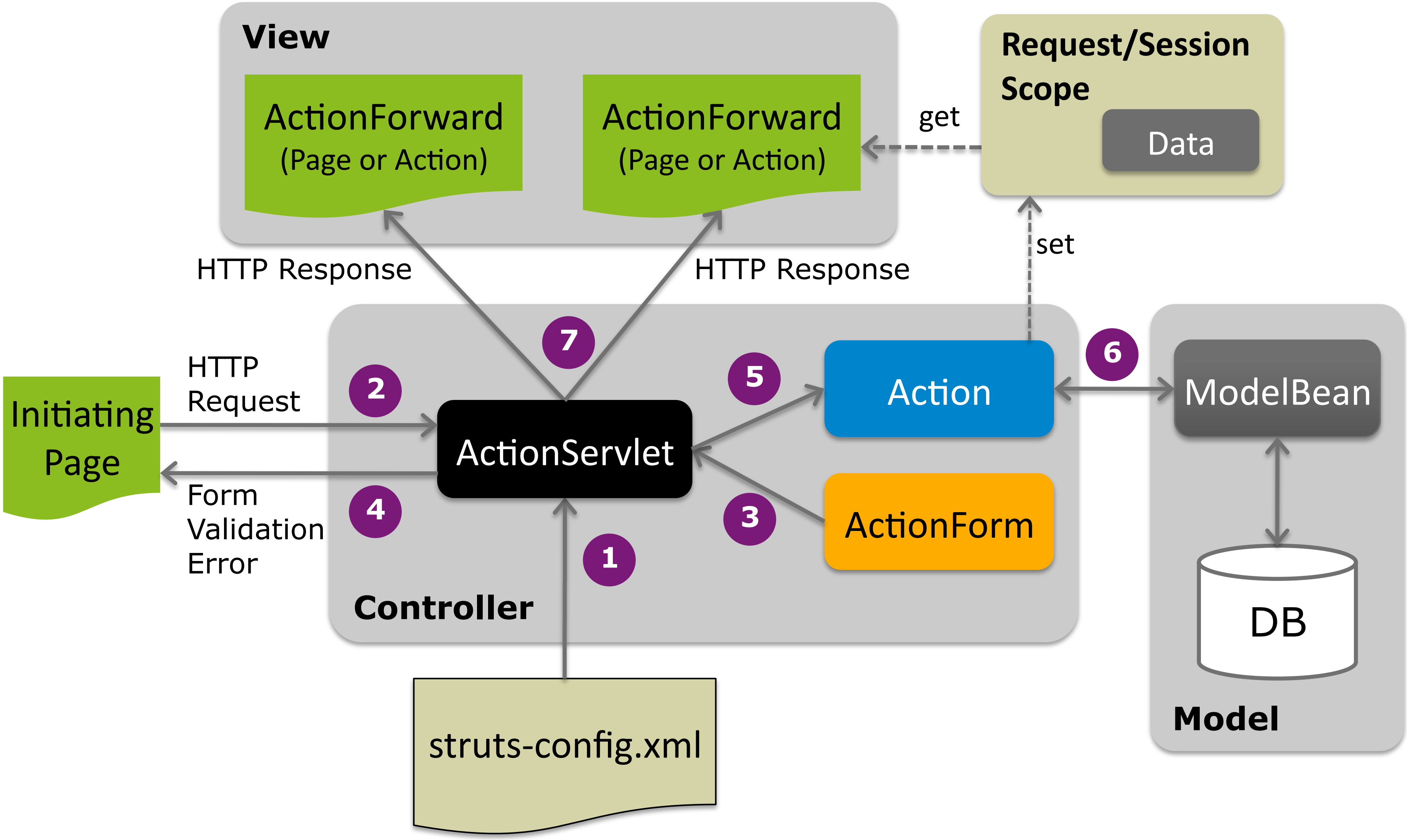
- **Black-Box Framework Implementing MVC**
  - Framework: reusable “partial” application
- Struts `ActionServlet` provides high level control of workflow (Part of Controller)
- You provide Beans and files to customize framework according to your application needs
  1. JSPs provide HTML presentation (View)
  2. `ActionForm` Beans “collect” form data (Part of Controller)
  3. `Action` Beans provide details of flow (Part of Controller)
  4. `struts-config.xml` declares Beans and JSPs

How To Develop Struts Applications

From 10 Miles High:

- Pass high-level control to `ActionServlet`
  - By appropriate URL mapping in `web.xml`
- Design “workflow” in diagrams and then code it in `struts-config.xml`
- Develop `ActionForm` Beans
- Develop `Action` Beans
- Develop Model Beans (not part of Struts)
- Develop HTML and JSP pages

Struts Single Request Processing



Struts Single Request Processing (cont'd)

1. When web app is loaded, `ActionServlet` parses `struts-config.xml` and associates URL paths with `Action` and `ActionForm` Beans
   - Location of `struts-config.xml` is given in `web.xml`
2. The user issues an HTTP request from an initiating page **P** to the `ActionServlet`

Struts Single Request Processing (cont'd)

3 The `ActionServlet` instantiates the `ActionForm` Bean associated with the HTTP request URL in `struts-config.xml`, and sets its properties using the HTTP request parameters (user-submitted data)

4 The `ActionForm` Bean validates its property values and if validation fails, `ActionServlet` responds with the initiating page **P** displaying appropriate error messages for the user to correct his/her form data

Struts Single Request Processing (cont'd)

5 If validation succeeds, the `ActionServlet` instantiates the `Action` Bean associated with the HTTP request URL in `struts-config.xml`, and calls its `execute` method passing as parameters the `ActionForm` Bean, the HTTP request and the HTTP response objects

Struts Single Request Processing (cont'd)

6 Within its `execute` method, the `Action` Bean instantiates/calls Model Beans, which open a connection to the database, execute SQL operations, and return sets of tuples

The `Action` Bean places the sets of tuples in the session so that JSP pages (View components) can access them

Struts Single Request Processing (cont'd)

7 The `Action` Bean returns to the `ActionServlet` one of the `ActionForward`s with which the HTTP request URL is associated in `struts-config.xml`

An `ActionForward` is a possible outcome of the `Action` Bean and represents either an JSP/HTML page or another `Action` that will be the response to the user's request

Upon receiving the `ActionForward`, the `ActionServlet` responds to the user's request with the corresponding JSP/HTML page or `Action`

Install Struts

- We will use Struts 1.3 for Phase 2 of the project
  - Struts 2 will be covered later on and will not be used for the project
- Download **struts-1.3.10-all.zip**
- Struts is only a package containing:

  \doc, \src, \lib, **\apps**
- Within \apps is a set of *.war files
  - struts-blank-1.3.10.war
  - struts-examples-1.3.10.war
  - struts-cookbook-1.3.10.war

Struts Examples

- To play with Struts examples:
  - Copy struts-cookbook-1.3.10.war under \webapps
  - Access http://localhost:8080/struts-cookbook-1.3.10/
- To play with more Struts examples:
  - Copy struts-examples-1.3.10.war under \webapps
  - This automatically deploys a new web app directory
  - Access http://localhost:8080/struts-examples-1.3.10/
- To start your own Struts application:
  - Copy struts-blank-1.3.10.war under \webapps
  - Rename \struts-blank-1.3.10 to \your_app_name