JUGS
Work In Progress Seminar

JSP - More than Dynamic Pages: When and Why to Use It

Presented by:

Paul Giotta
Tiberiu Füstös

22 May 2000

Cambridge Technology Partners

Presentation Overview

Part 1

- Project Requirements
- Solution Landscape
- Introduction to JSP
  - Programming features
  - Application server features
- JSP and Servlets
- JSP and EJB
- Container Vendors
- Platform Selection Criteria

Part 2

- Design Drivers
- Application Model
- Model/View/Controller Approach
- Simple Forms
- Complex Forms
- OO Solution
- Deployment
- Security
- Future developments

Project Background and Requirements

- Client and Project are Confidential
- Security is Critical
- Multi-Lingual

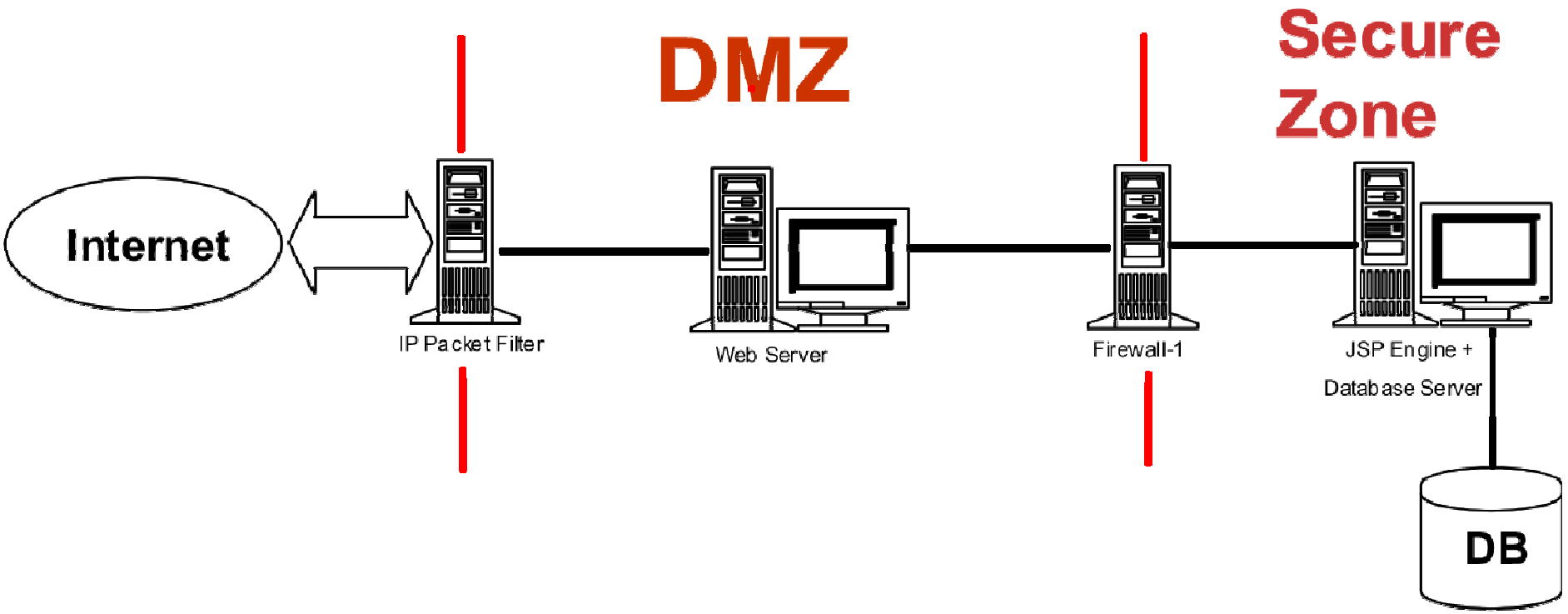
Phase 1:

- Bring client's current business processes to the web ASAP
- Very short design and development cycle
- Small amount of custom functionality
- ***Maximize reuse in phase 2***
- ***Keep it simple***

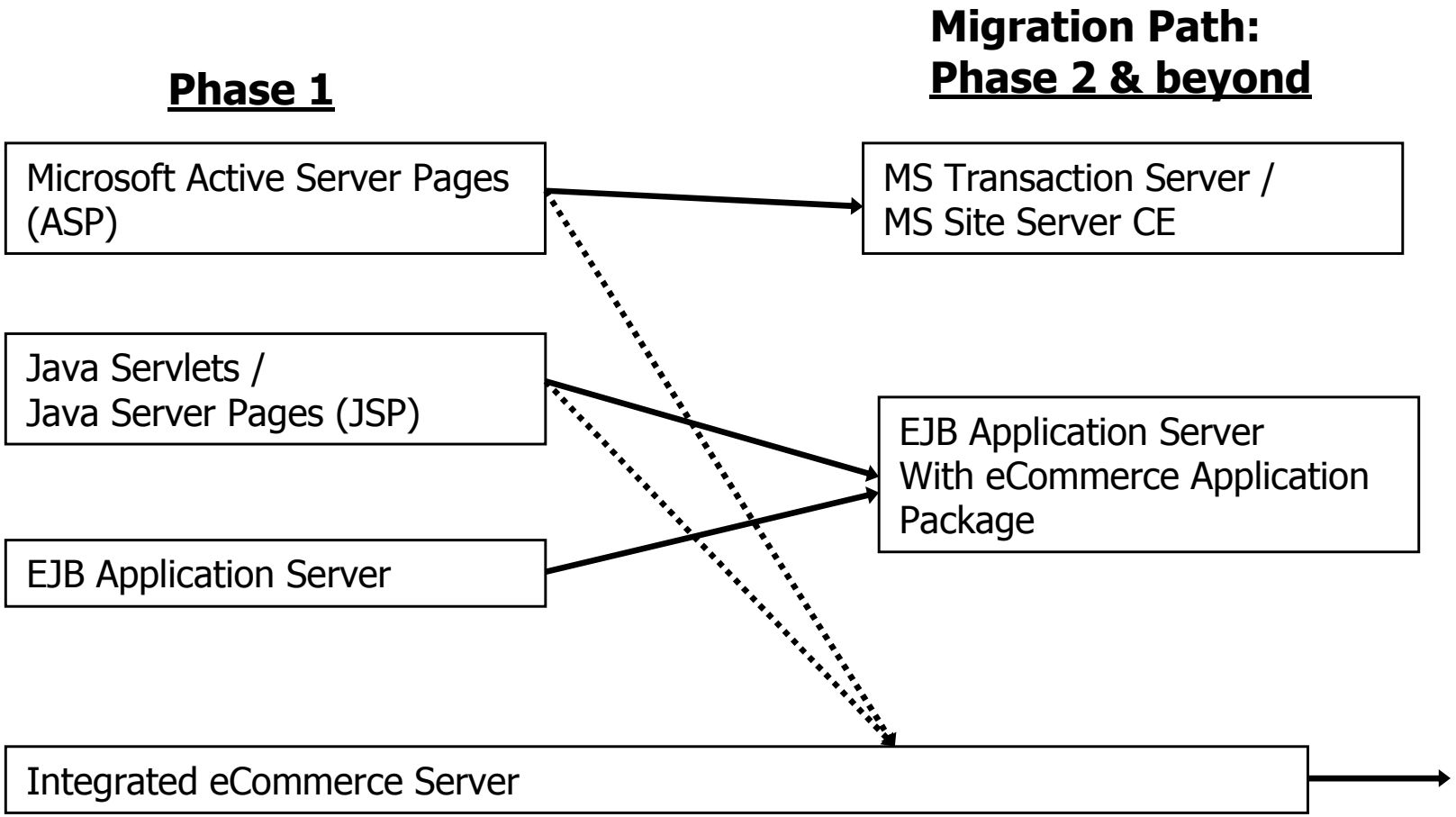
Phase 2:

- High volume eCommerce site
- Lots of standard functionality:
  - Catalog, Shopping Cart, Payment, Personalization, Cross Selling, etc.
- Specific requirement not yet determined

Network Architecture



Solution Landscape



Intro to JSP: Page Elements

- Standard Tags
- 'Use Bean'
- Session Management
- Alternate Languages
- Custom Tags

```
...<BODY>
<%@ page language="java"
         import="com.acme.app.*, java.util.*" %>

<jsp:useBean id="hist"
             scope="session"
             class="com.acme.app.histBean" />

<%
    Iterator itOrders = hist.getOrderList().iterator();
    while (itOrders.hasNext()){
      Order odr = (Order)itOrders.next();
%>

      <TR>
      <TD>Autrags-Nr.: </TD>
      <TD> <%= odr.getRenderer("odrNum").disp()%> </TD>
      <TD>Liefer-Datum:</TD>
      <TD> <%= odr.getRenderer("dvDt").disp()%> </TD>
      </TR>

      <%
       }
      %>
</BODY></HTML>
```

Intro to JSP: App Server Features

- Applications
  - 'Virtual Server'
  - One of 4 levels of scope for a bean
- Deployment
  - Deployment Descriptors
  - WAR file
- Security
  - Forms based
  - Defined at Application Level
- JSP Server remotlely connected to Web Server
  - Dynamic pages must not be in DMZ :-)
  - Some support for Load Balancing / Fault Tolerance

JSP and Servlets

- JSP extends Servlet API
- Many features are actually inherited from Servlet specification
- JSP spec requires implementations to compile JSP's into Servlets
- Strong relationship between versions of Servlet & JSP specs:

  Servlet 2.1 ⟷ JSP 1.0

  Servlet 2.2 ⟷ JSP 1.1

JSP and EJB

- JSP 1.1 / Servlet 2.2 define the equivalent of an EJB *Web Container*
- JSP applications should be directly portable to EJB containers

	JSP	EJB
Web presentation layer	JSP Tags	JSP Tags
Application Logic	Java Beans	Session Beans
Persistence	No direct support (DIY): Beans w/JDBC	Session Beans w/JDBC Enitiy Beans w/CMP (DIY)
Deployment	WAR file Deployment Descriptors	WAR file Deployment Descriptors

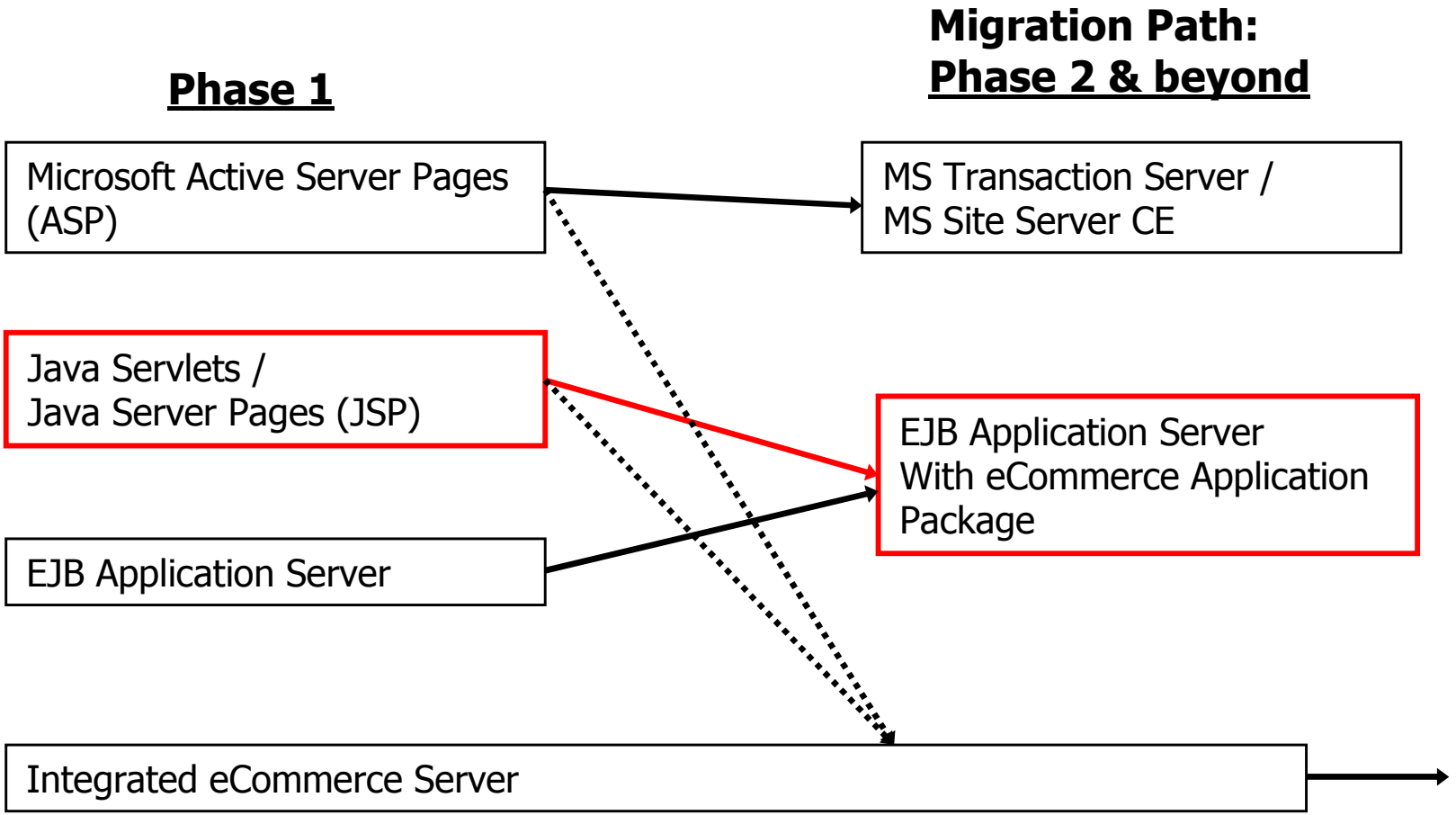
JSP Containers: Vendors & Pricing

Product	Approx. Price (USD)	Comments
Allaire JRun	Base version: free Pro version: $600 per processor Pro Unlimited: $2000 per machine	Largest installed base
Caucho Resin	Free for non-commercial use $500 per Server $2500 with premium support	Most features Open Source
Jakarta Tomcat	Free	Open Source Offical Reference Implementation Part of the Apache project

Platform Selection Criteria

- Security
- Standards Compliance
- Upward compatibility w/ EJB
- Delay decision for expensive, transactional, eCommerce platform
- Quick implementation

Solution Landscape Revisited



Conclusions

- JSP defines an application server that is well suited to projects of small to medium complexity.

- JSP containers are an excellent starting point for organizations that want to pursue EJB development, but are not ready to commit to the cost and learning curve of a full EJB product.

JUGS
Work In Progress Seminar

JSP - More than Dynamic Pages: When and Why to Use It

Part II

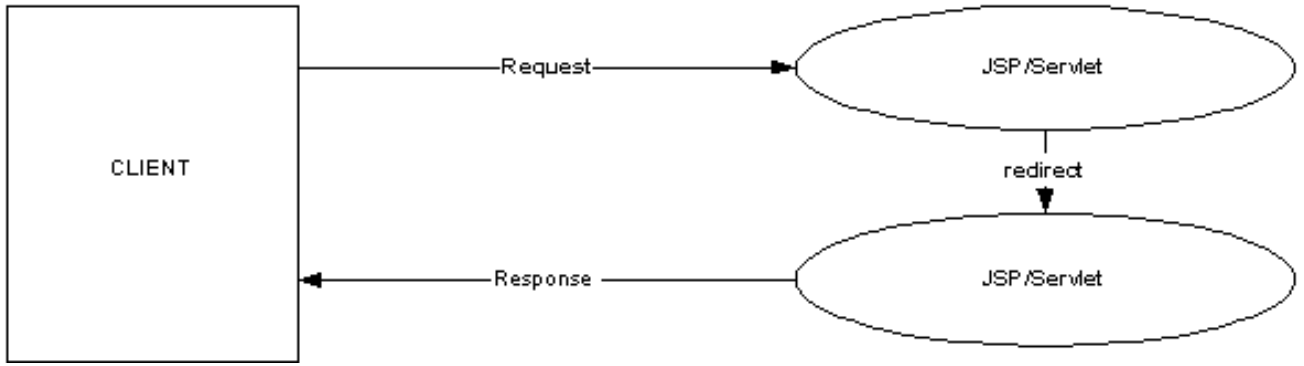
Cambridge Technology Partners

Design Drivers

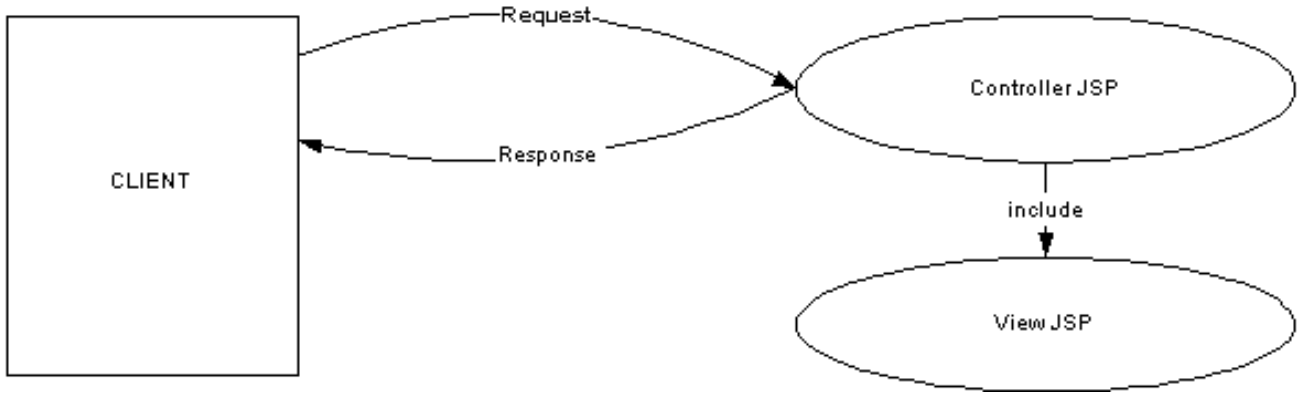
- Multi-disciplinary team (business, technology, creative, cognitive)
- Easy maintenance of language specific elements
- No application processing in pages with visual elements (easier maintenance)
- Run-time forms generation (arbitrary number of forms elements)

Application Model

- "Model 2"



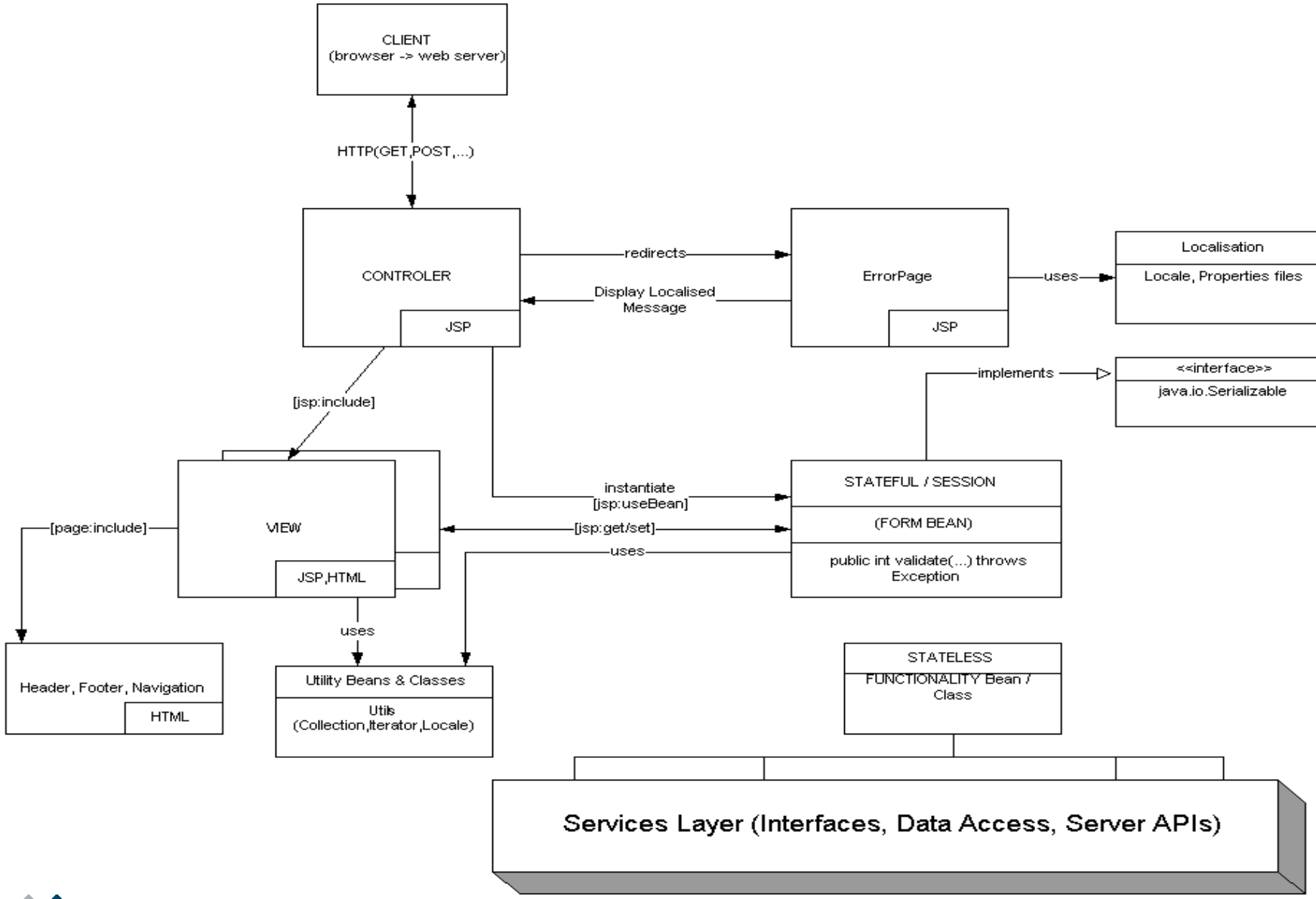
- "Include Request"



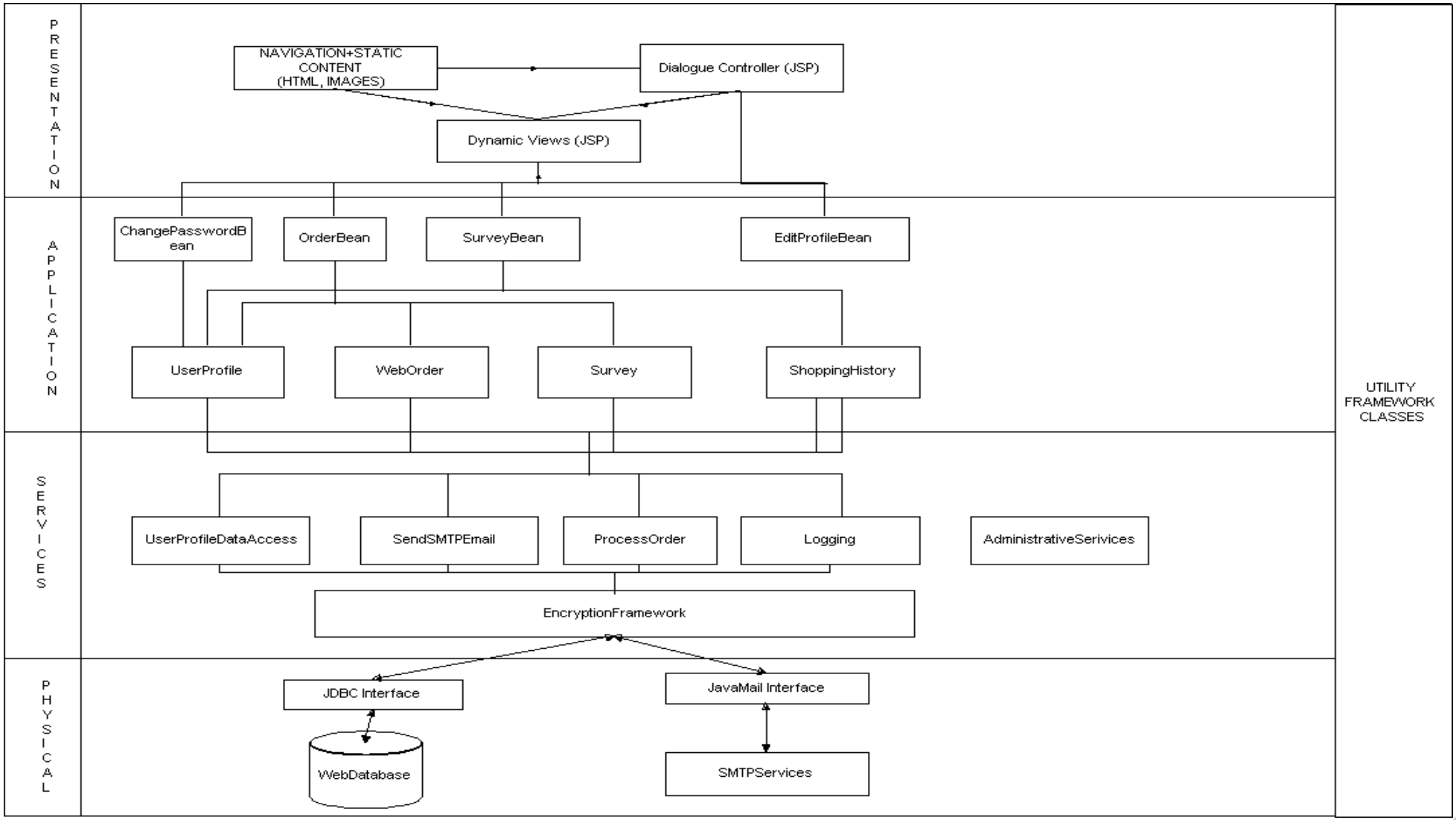
Why a JSP Controller?

- Full JSP tags support
- Speed of development / test cycles (uniform process as opposed to serlvet / JSP)
- Consistent deployment
- Same security as for all the JSPs
- Essentially it is a servlet generated and compiled by the JSP run-time

Detailed Model/View/Controller Approach



High Level Application Blueprint

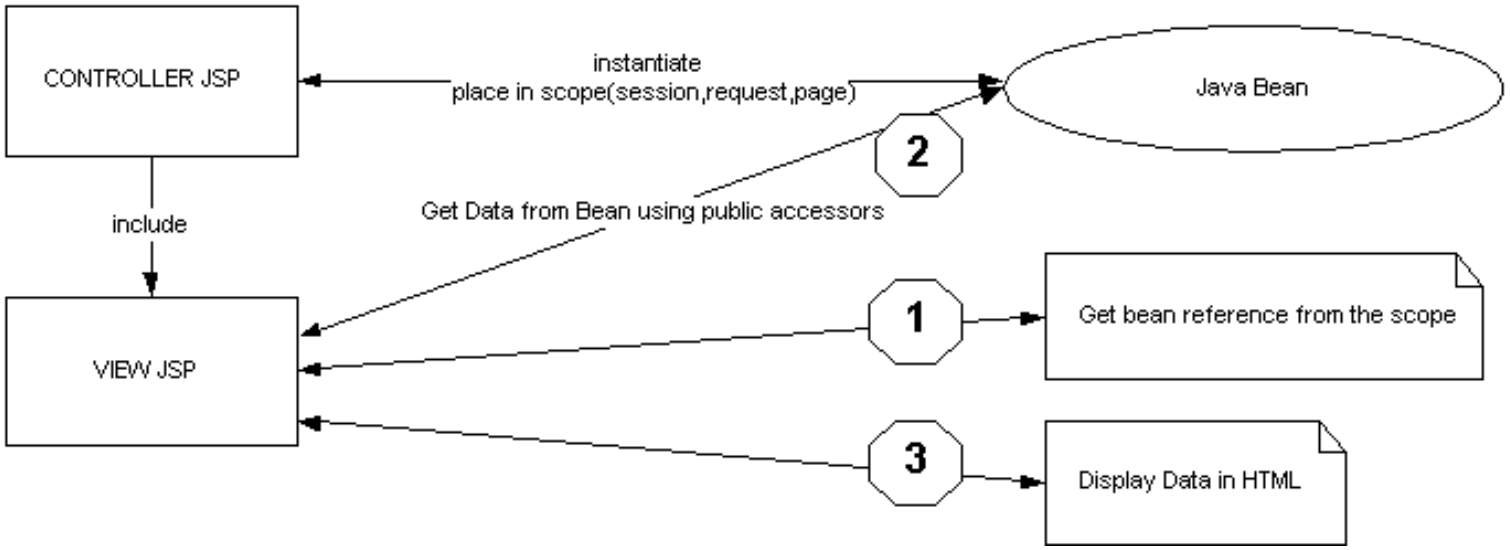


Simple-Forms Processing

- Defining a "simple" form:
  - "a simple form is an HTML form which can be designed by a creative designer using common productivity tools" (DreamWeaver, NetObjects Fusion, GoLive!)
- Main differentiation:
  - all the form elements are known at implementation/design time and their structure and number are well defined
- Example:
  - "login" form: contains 2 fields: username (type "text") and password (type "password")

Simple-Forms Processing Model

- On entering the page:



- On user action (processing done by the controller):
  - get and dispatch user actions
  - <jsp:setProperty name="bean" property="*" />
  - call methods of the bean (validate(), process() etc.)
  - handle errors

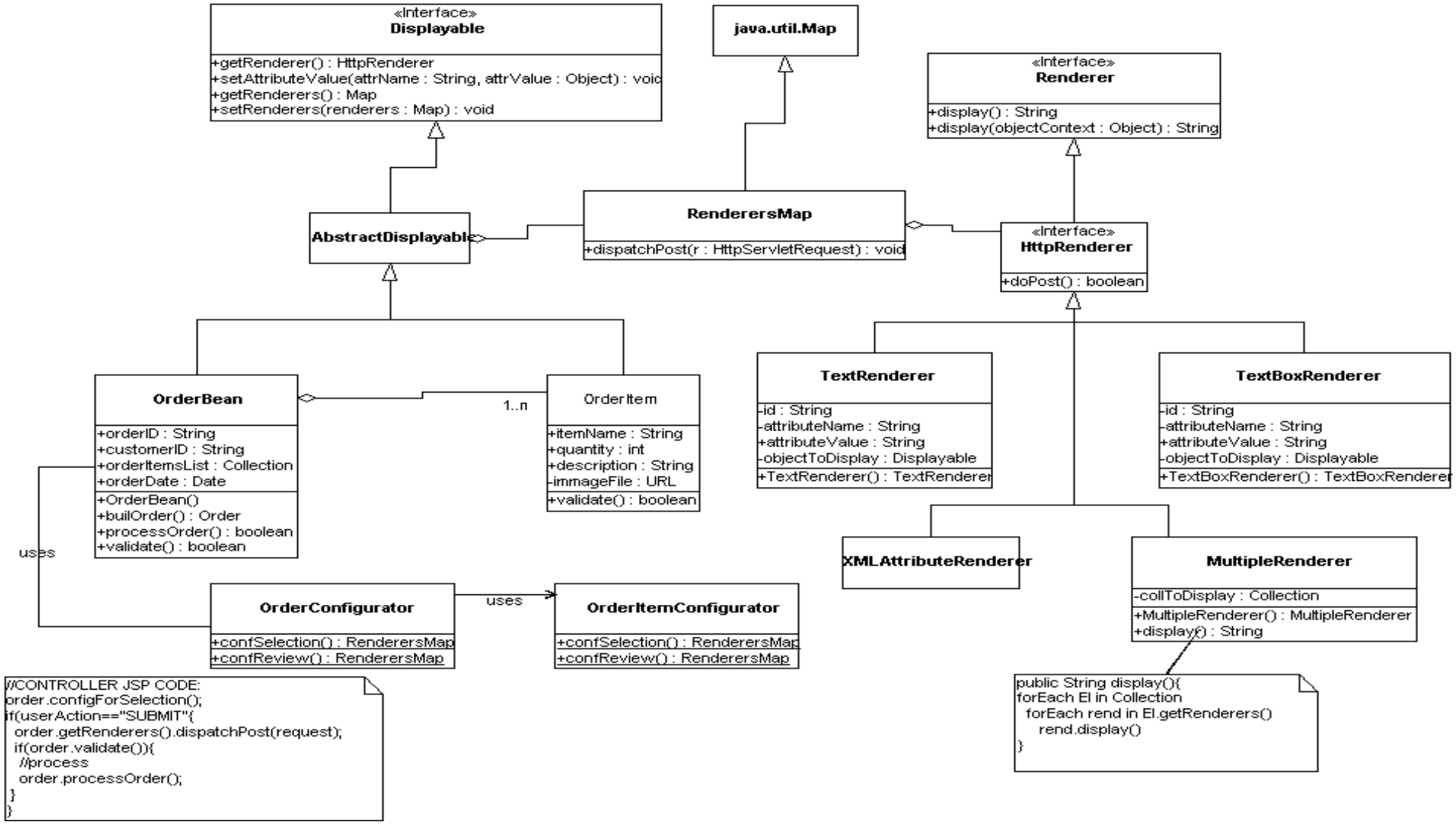
Complex Forms

- Main characteristics:
  - the number and/or type of form elements are not known at design time
- Example:
  - web-based ordering from arbitrary number of personalized items
- Solutions:
  - "procedural" solutions - query the number of items and build views which extract data from the objects and link to other views [...]
  - "OO" solutions - let the objects display themselves. They know best their needs!

Used Solution: OO

- An adapted PAC/MVC solution
  - Presentation/Abstraction/Controller from: *Pattern Oriented Software Architecture: A System of Patterns, Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, and Michael Stal (John Wiley & Sons, 1996)*
- Each object from the abstraction layer which needs an UI implements a common interface.
- The representation of the objects' attributes is delegated to "renderers"
- Event-style messages flow directly from the presentation layer to the model layer
- Factory classes can be used to assign different "renderers"
- The UI interacts with the renderers and a front bean

Object Model



Cambridge Technology Partners

Deploying the Web-Applications

- Deployment descriptor: makes the application independent by the server administrator
- Packaging: .WAR (.JAR with "W" from Web)
- What can be specified at deployment:
  - session configuration
  - servlet / JSP definitions and mappings
  - MIME type mappings
  - welcome file list
  - security
  - error handling

Security

- The specification covers the basics for autentication, authorization (access control), data integrity, confidentiality
- Most important: support for *declarative security* as well as programmatic security
- Not covered by the specification (to be checked when evaluating products):
  - authentication frameworks
  - back-end interoperability with other authentication protocols
  - cross-application authentication and trust model (this is specified by the J2EE spec.)

Specifying Security Constraints

- Security is "container-tracked" and each application can specify its constraints in the deployment descriptor (web.xml)
- Sample descriptor:

```
<!-- ... -->
<security-constraint>
  <web-resource-collection>
    <url-pattern>/home/my_application/*.jsp</url-pattern>
  </web-resource-collection>
  <auth-constraint>
     <role-name>administrator</role-name>
  </auth-constraint>
  <user-data-constraint>
     <transport-guarantee>CONFIDENTIAL</transport-guarantee>
  <user-data-constraint>
</security-constraint>
<!-- ... -->
```

Future Developments

- JSP /Servlet - only implementations come closer to a full “web-container” implementation required by J2EE
- Extended support for internationalization in the framework (“getResource” in the ServletContext)
- Extending and implementing “custom” tags for the large base of developers used to ColdFustion type of RAD tools

References

- Links:
  - http://jakarta.apache.org/tomcat/index.html
  - http://www.caucho.com
  - http://www.allaire.com/Products/JRun/
  - http://www.javaworld.com/

- Books:
  - Ayers et al.,*Profressional Java Server Programming*, (Wrox Press Ltd., 1999)
  - Buschmann, Frank, Regine Meunier, Hans Rohnert, Peter Sommerlad, and Michael Stal, *Pattern Oriented Software Architecture: A System of Patterns*, (John Wiley & Sons, 1996)
  - Gamma, Erich, Richard Helm, Ralph Johnson, and John Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*, (Addison-Wesley, 1995)