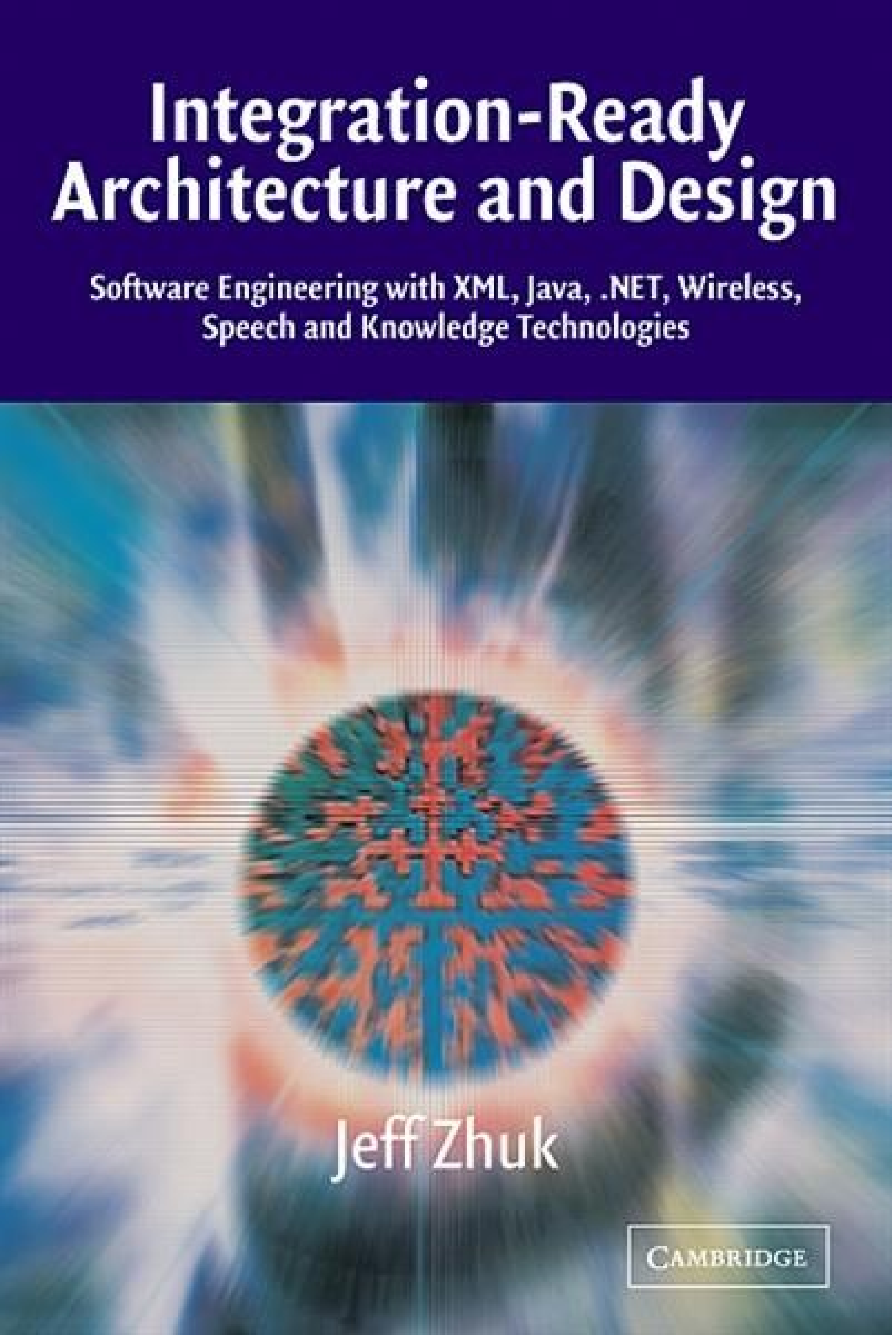

Integration-Ready Architecture and Design
Software Engineering with XML, Java, .NET, Wireless, Speech and Knowledge Technologies
Jeff Zhuk
CAMBRIDGE

JEE Web Applications

Jeff Zhuk

From the book and beyond

"Integration-Ready Architecture and Design"

Cambridge University Press

JavaSchool.com

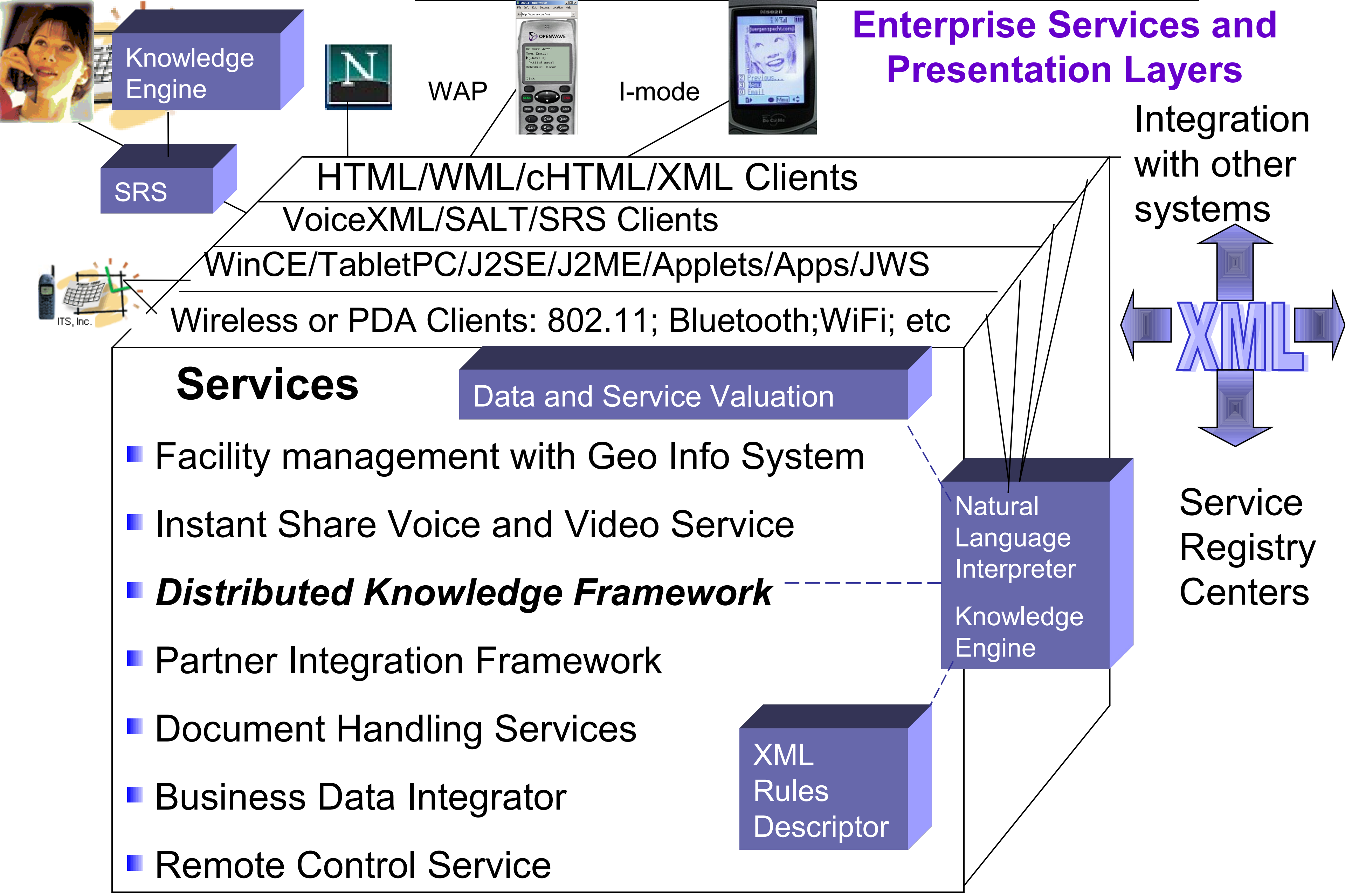
Software Engineering With

XML, Java, .NET, Wireless,

Speech and Knowledge

Technologies

Enterprise Services and Presentation Layers


Knowledge Engine
SRS
WAP
I-mode
Integration with other systems
HTML/WML/cHTML/XML Clients
VoiceXML/SALT/SRS Clients
WinCE/TabletPC/J2SE/J2ME/Applets/Apps/JWS
ITS, Inc.
Wireless or PDA Clients: 802.11; Bluetooth;WiFi; etc
XML
Data and Service Valuation
Natural Language Interpreter
Knowledge Engine
Service Registry Centers
XML Rules Descriptor

Services

- Facility management with Geo Info System
- Instant Share Voice and Video Service
- ***Distributed Knowledge Framework***
- Partner Integration Framework
- Document Handling Services
- Business Data Integrator
- Remote Control Service

Multi-tier Enterprise Architecture

Client Container

Browser or mail client

Client Device

With Embedded Application

802.11

Bluetooth

Mail Server

Web Server

ASP.NET

Or

JSP/Servlet

Dynamic Content Engines

Service Container

(App Server)

Message Queue

Business Services

Connectors

Device and data drivers

XML rules and descriptors

Data and Remote Systems

LDAP RDBMS DS, NS

Control Devices

Legacy Apps

A multi-tier Open Enterprise architecture

- A multi-tier open Java based enterprise architecture is built as a set of extensible services-frameworks with ability to add/customize services run-time
- Tier 1 – Client requests services via XML based service API

- Client types: 1) partner application running on a workstation, 2)Web Browser with Java™ Applet , 3)Wireless device with embedded WML browser or VoiceXML interpreter, 4)Java card technology device, etc.
- Tier 2 – Web Container with J2EE Servlet and JSP engines where servlet is responsible for session tracking and request distribution, and JSPs provide presentation layer back to the client. Tier 2 can be considered as a communication tier that in the case of HTTP serves as a Web Container
- Tier 3 – worker beans providing services.
  Worker beans can be (not necessary) implemented as EJBs to gain advantage of security and transaction monitoring services provided by EJB containers.
- Tier 4 – Connectors to Data, Remote Systems, etc. (XML API to Tier5)
- Unified JNDI based approach is used for data integration describing data types, rules, and structure with XML descriptors. A master controller with XML based API is created to describe a set of operations on device controllers.

Java Enterprise Services

Naming and Directory
– allows programs to locate services and components through the Java Naming and Directory Interface (JNDI) API

Authentication
– enforces security by requiring users to log in

HTTP
– enables Web browsers to access servlets and JavaServer Pages (JSP) files

EJB
– allows clients to invoke methods on enterprise beans

JMS
- enables asynchronous processing with messaging services

JEE Implementations of Model-View-Controller (MVC) Design Pattern

Web Applications before MVC:
-Common Gateway Interface (CGI)

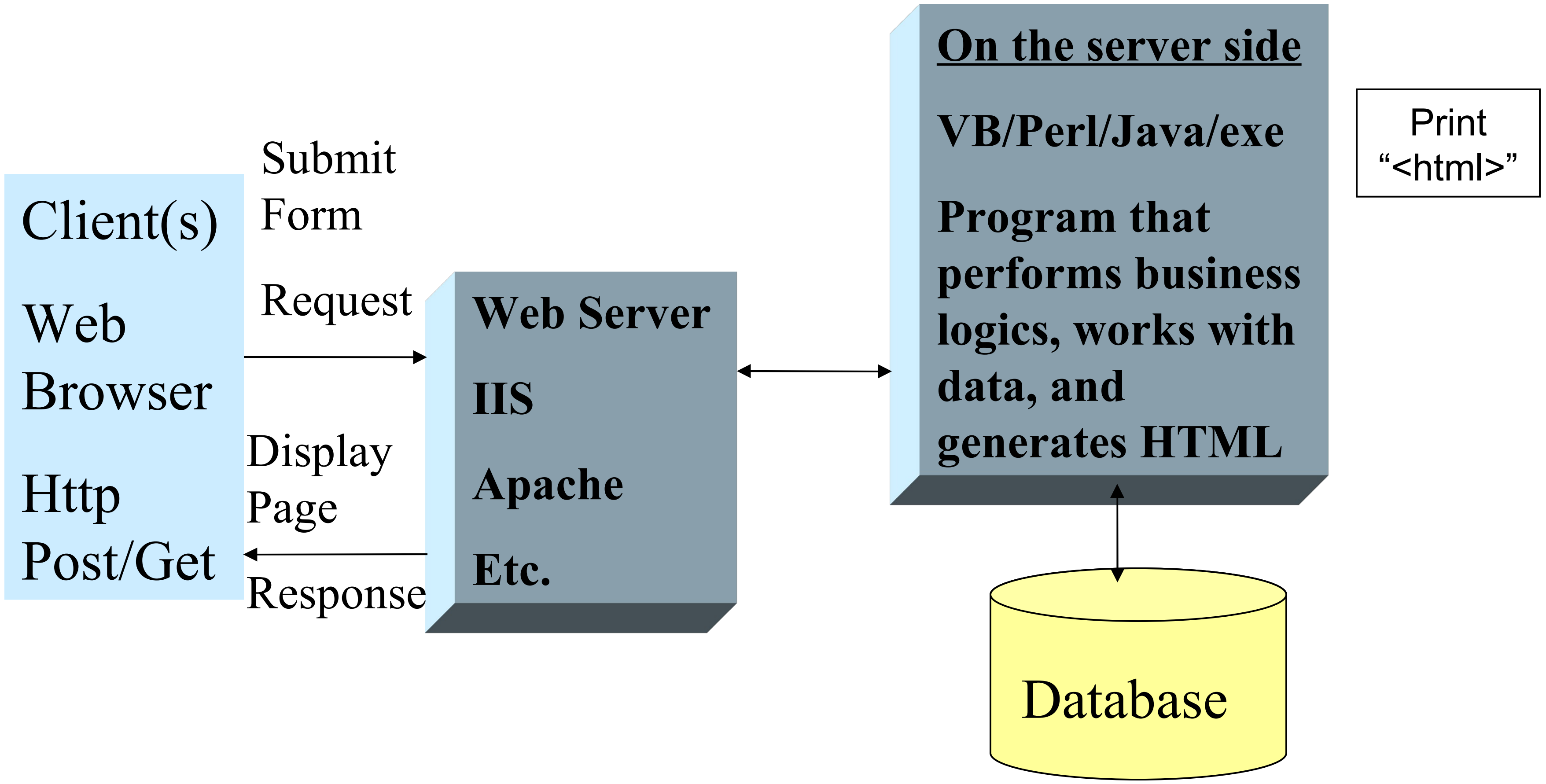
MVC Model 1 (Page-centric Architecture) JSP-to-JSP

MVC Model 2 (Servlet-centric Architecture

Open Source Web application frameworks: Struts and more

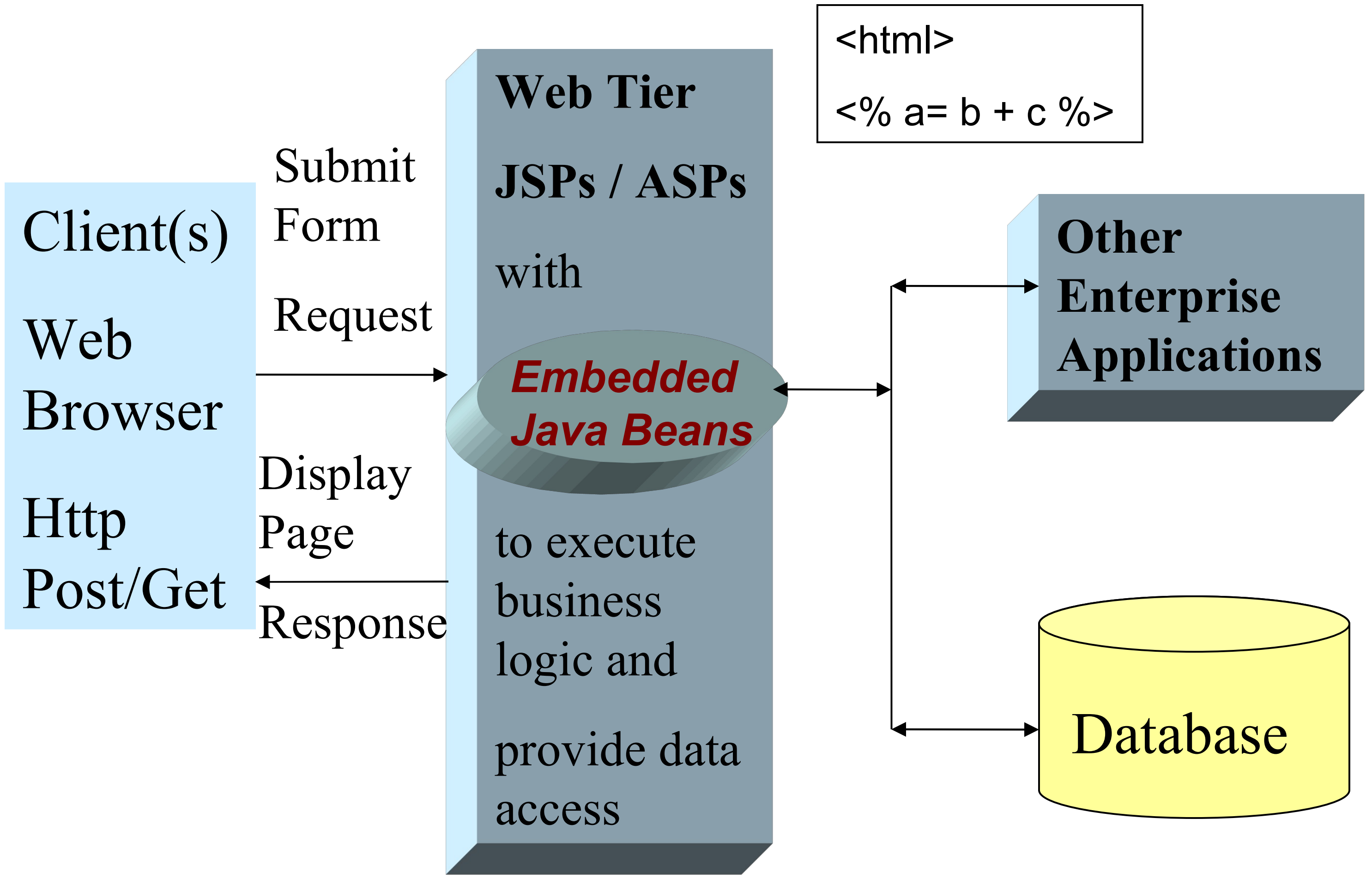
Standard-based Web application framework: Java Server Faces (JSR-127)

Common Gateway Interface (CGI)
A Mix of Presentation and Business Logics
With Generous Use of Resources



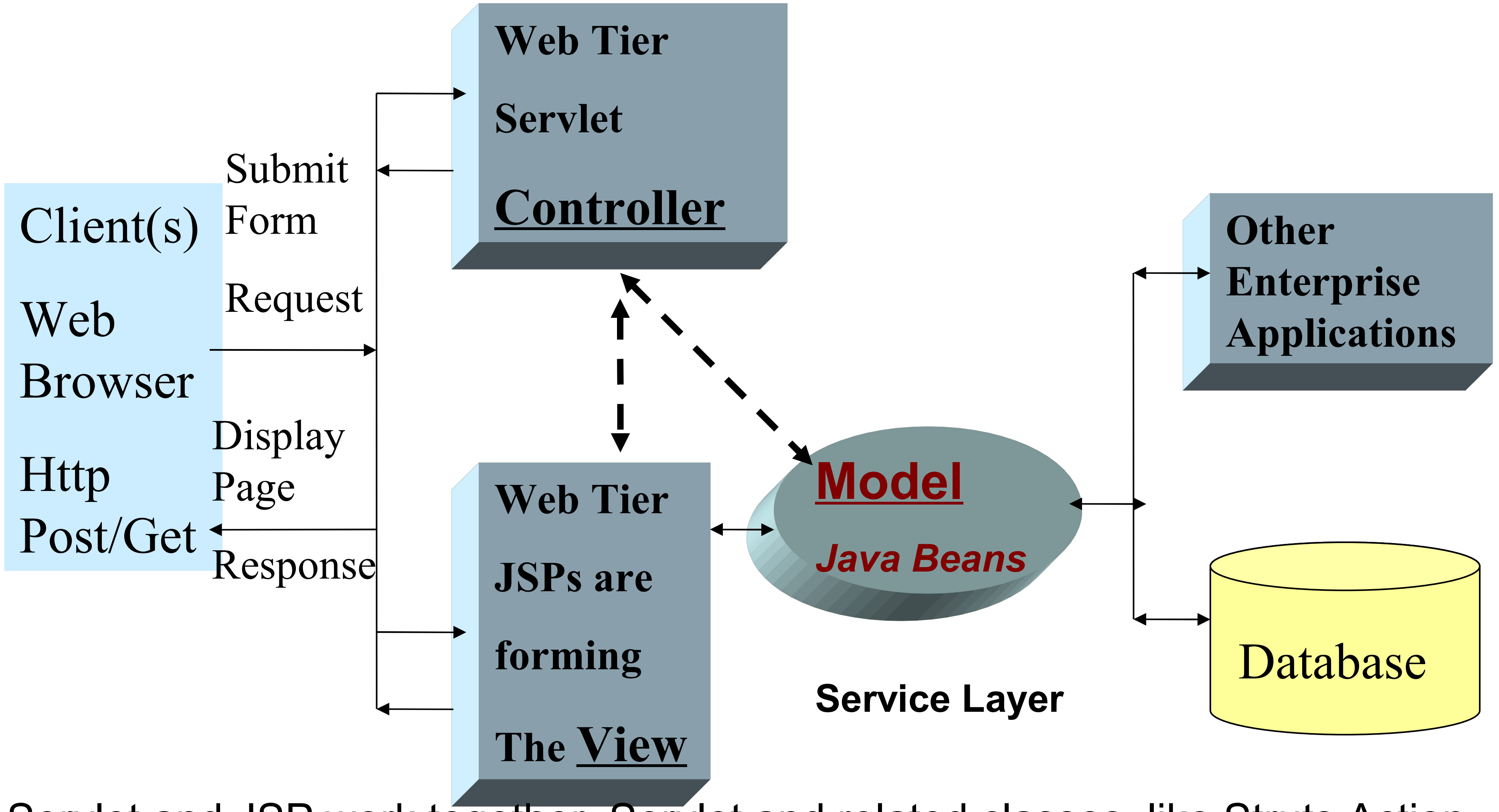
Each client request fires up a program (process) on the server side that performs business logic and sends a dynamic HTML page back to the client

MVC Model 1
Page-Centric Architecture

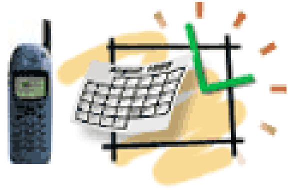


Interrelated JSP pages provide presentation, control, and business processing with scriplets and embedded Java beans encouraging “spaghetti” code in JSP.

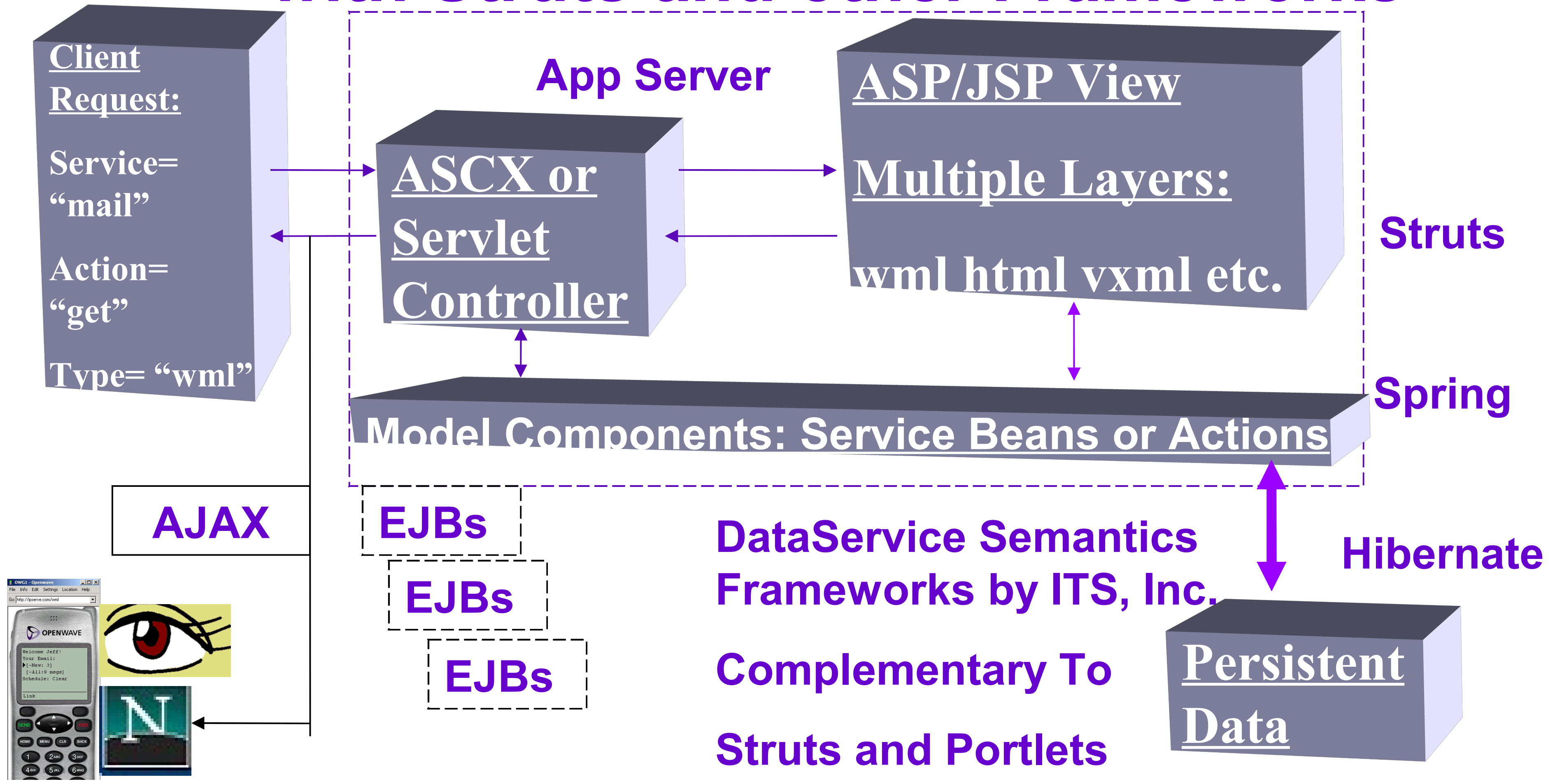
MVC Model 2 - Better Separation of Business and Presentation Layers



Servlet and JSP work together. Servlet and related classes, like Struts Action, control application logic and interact with services while JSP forms the presentation



Current Enterprise Web Applications with Struts and other Frameworks



DataService Semantics Frameworks by ITS, Inc.

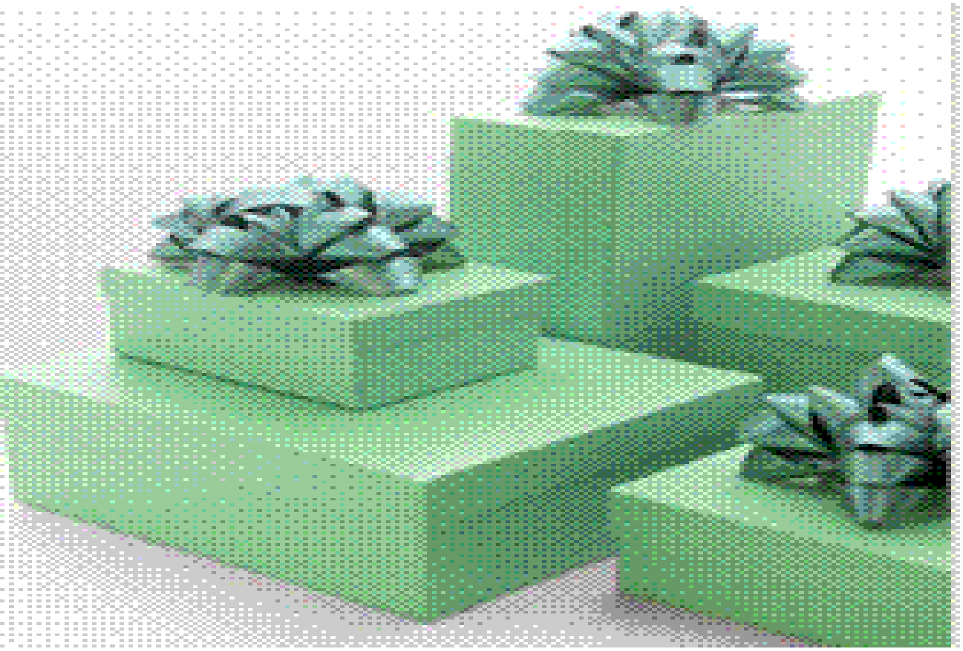
Complementary To

Struts and Portlets

MVC Design Pattern (J2EE/ASP.Net)
Multiple Presentation Factories
(HTML/WML/etc.)

Self-Healing Well Packaged Applications

Use existing frameworks to:



- Deliver basic operation statistics

- Monitor application health

- Validate application data

- Prepare application for work (Leave only DDL in your release notes, use app services to prepare data)

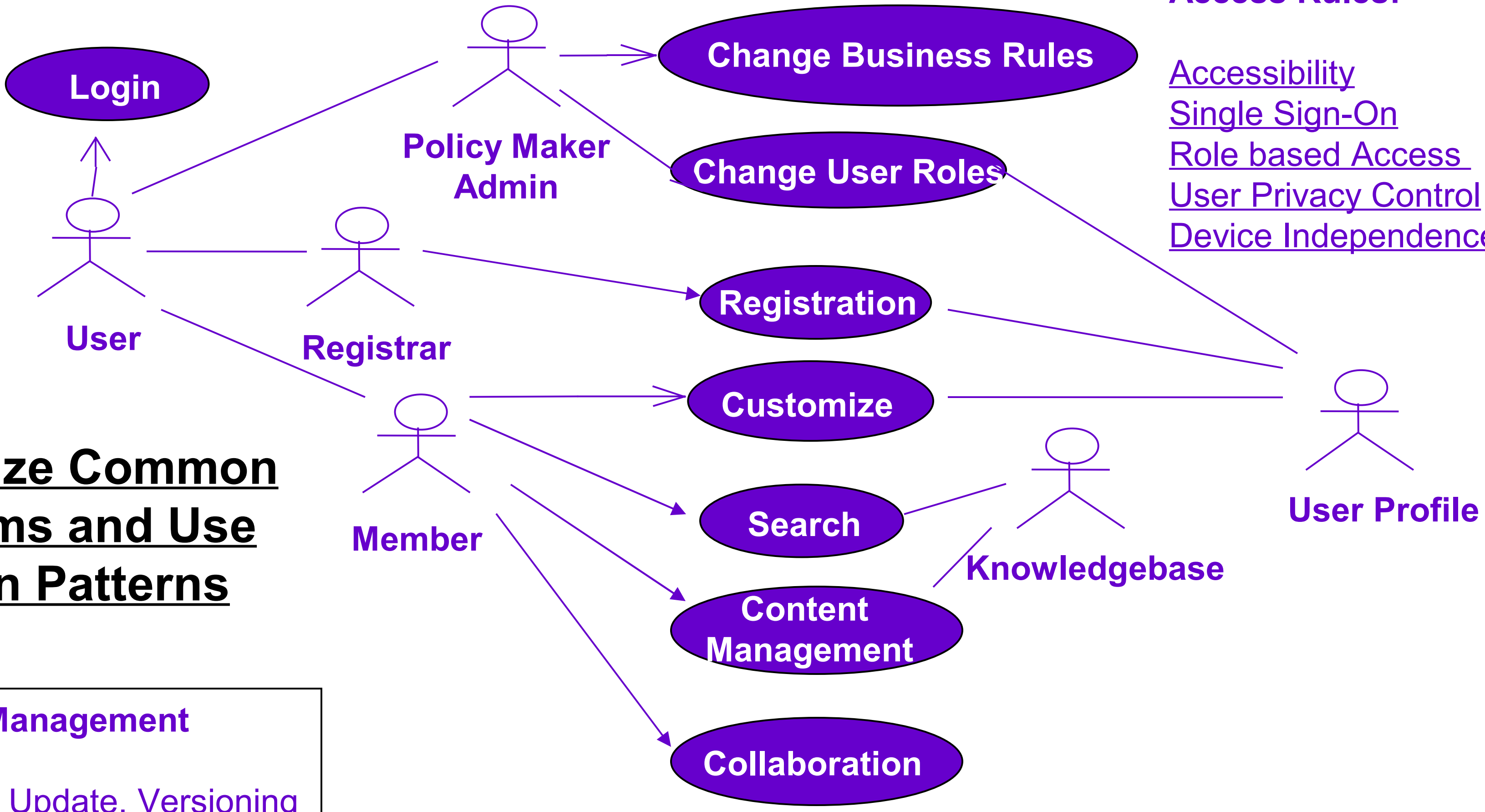
- Provide testing facilities

Provide standard ways for data exchange

Provide standard ways to configure applications

e-Business *e-Government* *e-Training*

Common Portal Use Cases



Access Rules:

Accessibility
Single Sign-On
Role based Access
User Privacy Control
Device Independence

Recognize Common Problems and Use Design Patterns

Content Management

Authoring, Update, Versioning
Forms, Permits, Applications
Scheduling events/facilities
Workflow Routing
Planning and Approval
Document/Photo Imaging
Task Tracking and Reporting
Data and Service Evaluation

Collaboration

Email
Conferencing
Instant Messaging
Privilege-based Data/Service Sharing

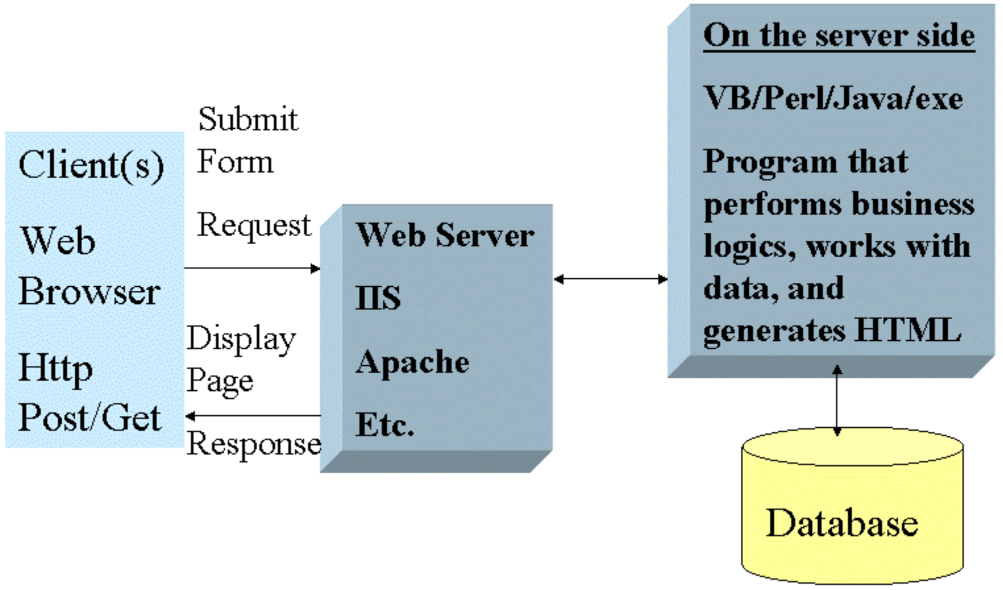
Search

Search for Data and Services
GIS (Maps and Routes)
Linking Related Cases
Content-based Subscription

Summary/Repetition Web Application Architectures

- **1 No MVC = CGI**
- 2 ?
- 3 ?
- 4 ?
- 5 ?

Common Gateway Interface (CGI) A Mix of Presentation and Business Logics With Generous Use of Resources



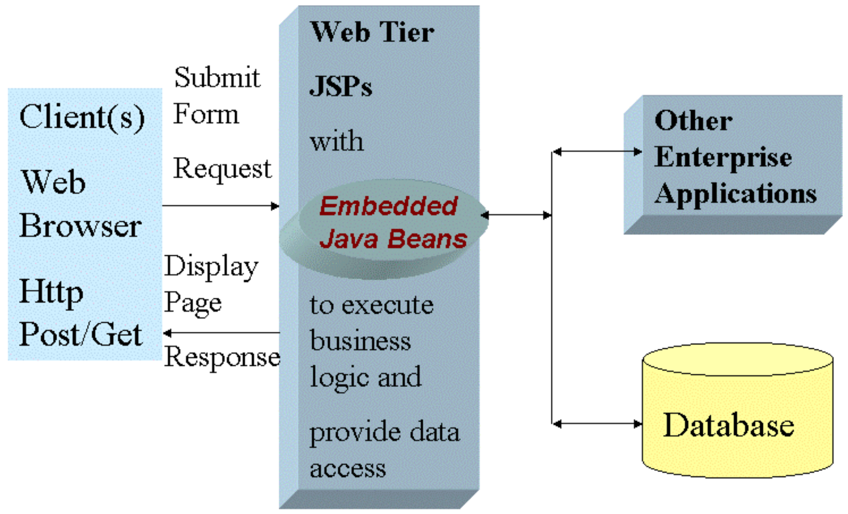
Each client request fires up a program (process) on the server side that performs business logic and sends a dynamic HTML page back to the client

Summary/Repetition
Web Application Architectures

- 1 No MVC = CGI
- 2 MVC Model 1 – Page-Centric Architecture

MVC Model 1
Page-Centric Architecture

- 3
- 4
- 5

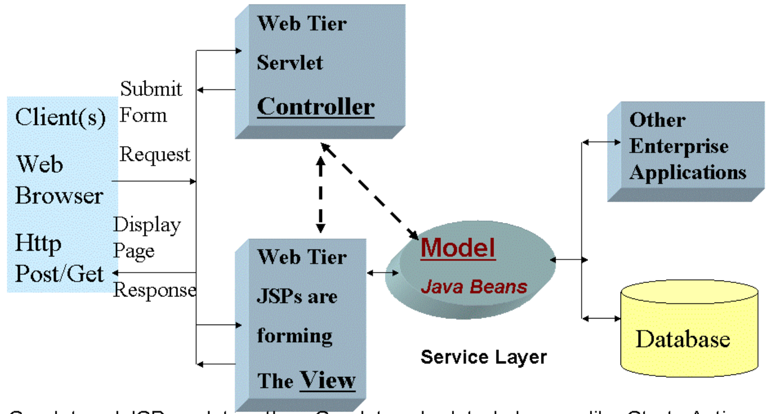


Interrelated JSP pages provide presentation, control, and business processing with scriplets and embedded Java beans encouraging "spaghetti" code in JSP.

Summary/Repetition Web Application Architectures

- **1 No MVC = CGI**
- **2 MVC Model 1 – Page-Centric Architecture**
- **3 MVC Model 2 - Better Separation of Business and Presentation Layers**
- 4 ?
- 5 ?

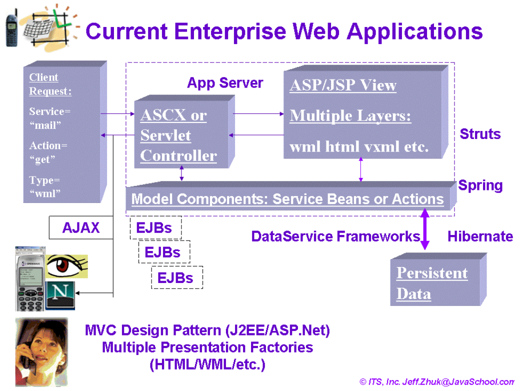
MVC Model 2 - Better Separation of Business and Presentation Layers



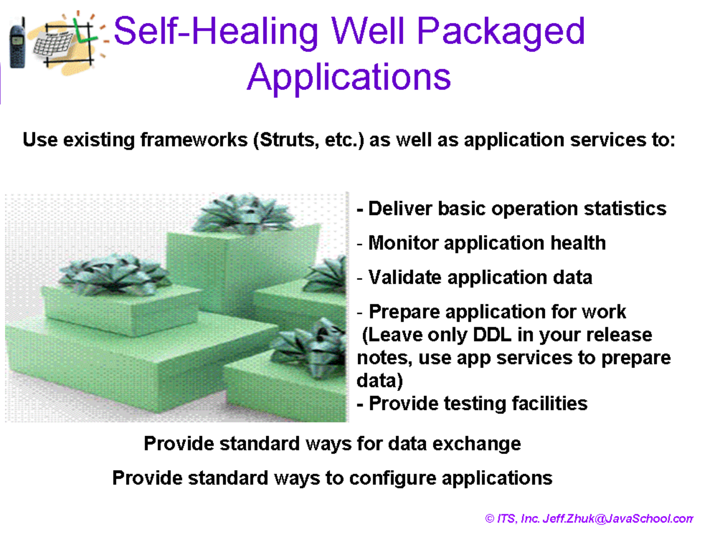
Servlet and JSP work together. Servlet and related classes, like Struts Action, control application logic and interact with services while JSP forms the presentation

Summary/Repetition Web Application Architectures

- **1 No MVC = CGI**
- **2 MVC Model 1 – Page-Centric Architecture**
- **3 MVC Model 2 - Better Separation of Business and Presentation Layers**
- **4 Struts and Other Frameworks reduce generic code**
- 5 ?


Current Enterprise Web Applications
Client Request:
Service= "mail"
Action= "get"
Type= "wml"
App Server
ASCX or Servlet Controller
ASP/JSP View
Multiple Layers:
wml html vxml etc.
Struts
Spring
Model Components: Service Beans or Actions
AJAX
EJBs
EJBs
EJBs
DataService Frameworks
Hibernate
Persistent Data
MVC Design Pattern (J2EE/ASP.Net)
Multiple Presentation Factories
(HTML/WML/etc.)
© ITS, Inc. Jeff.Zhuk@JavaSchool.com

Summary/Repetition Web Application Architectures



Self-Healing Well Packaged Applications

Use existing frameworks (Struts, etc.) as well as application services to:

- Deliver basic operation statistics
- Monitor application health
- Validate application data
- Prepare application for work (Leave only DDL in your release notes, use app services to prepare data)
- Provide testing facilities

Provide standard ways for data exchange

Provide standard ways to configure applications

- 1 No MVC = CGI
- 2 MVC Model 1 – Page-Centric Architecture
- 3 MVC Model 2 - Better Separation of Business and Presentation Layers
- 4 Struts and Other Frameworks reduce generic code
- 5 Self-Healing Well Packaged Applications