机关事业单位养老保险管理信息系统

技术培训—系统架构




机关事业单位养老保险管理信息系统构架组

培训简介

- **培训内容**
  - 机关保系统架构相关技术知识
- **培训目标**
  - 了解机关保的系统架构
  - 能够对机关保系统架构进行定制扩展
- **适用对象**
  - 本地化项目的系统架构师、技术人员
- **学员要求**
  - 熟悉JavaEE的相关技术与架构
  - 熟悉Struts、Spring、Hibernate等技术
  - 理解AOP等概念

目录

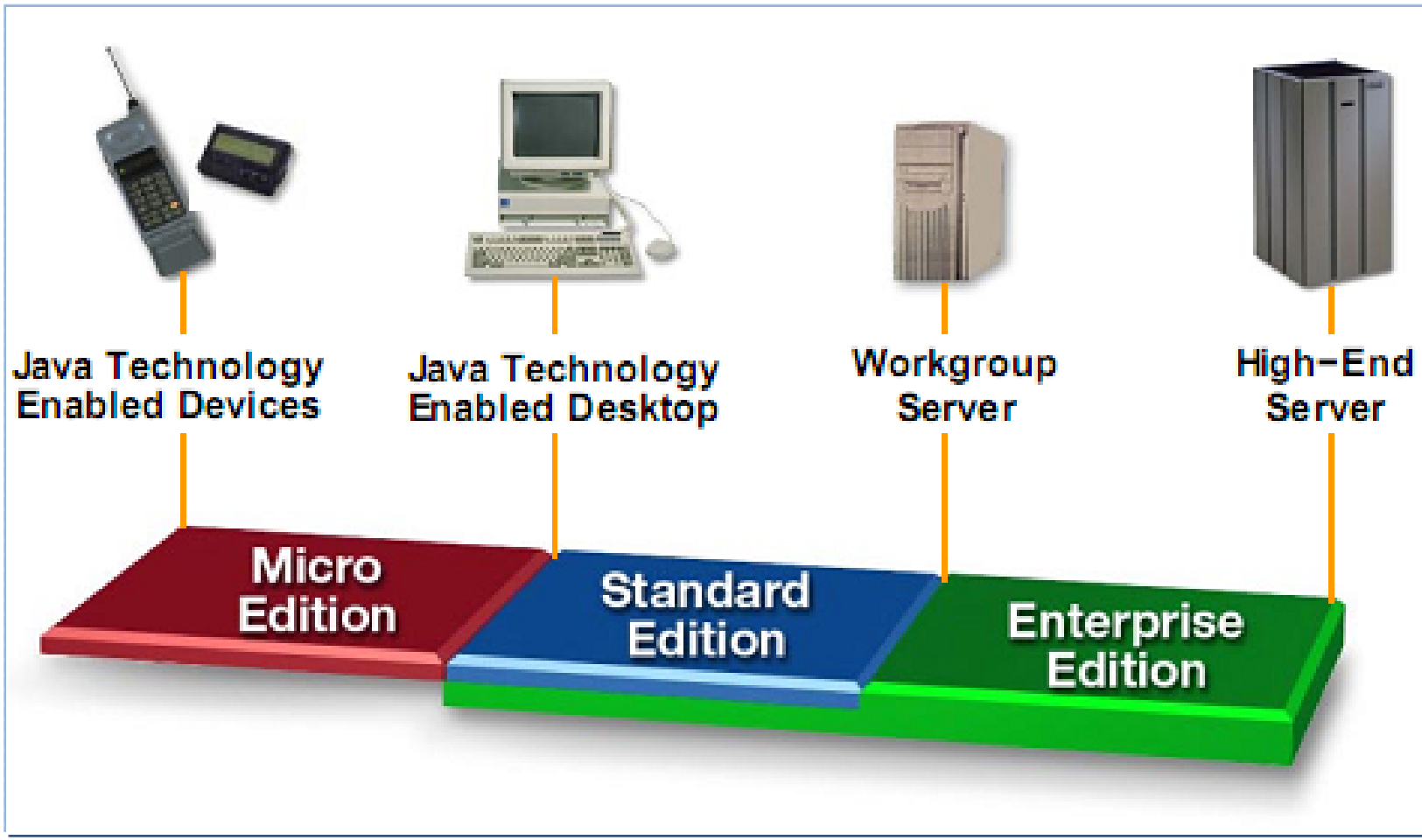


基础知识

基础知识

- JavaEE基础
- Struts框架
- Spring框架
- Hibernate框架
- 浏览器界面技术

JavaEE平台体系


Java Technology Enabled Devices
Java Technology Enabled Desktop
Workgroup Server
High-End Server
Micro Edition
Standard Edition
Enterprise Edition

什么是JavaEE

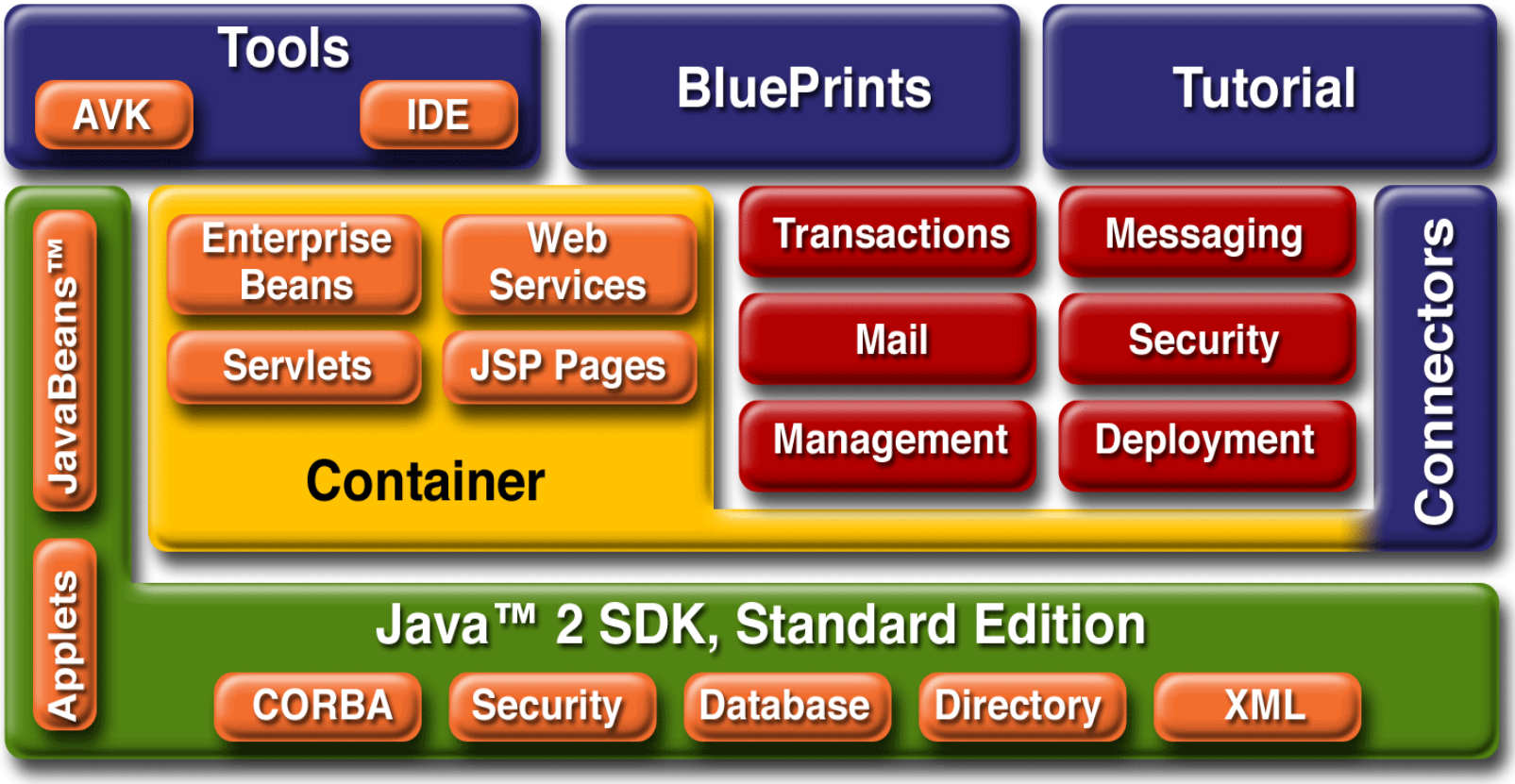
- **企业版**
  - 标准、开放的基础平台
  - 用于开发、部署、管理企业应用
  - 多层结构
  - 可使用Web
  - 以服务器为中心
- **Open and standard based platform for developing, deploying and managing n-tier, Web-enabled, server-centric enterprise applications**

JavaEE概念

- **JavaEE**
  - 以JavaSE为基础平台
  - 一套规范
  - 加上企业软件提供者的产品
- **JavaEE的交付内容**
  - JavaEE规范（Platform Specification）
  - 参考实现(Reference Implementation)
  - 兼容性测试套件（Compatibility Test Suite）
  - JavaEE蓝图（JavaEE BluePrints）

JavaEE平台服务

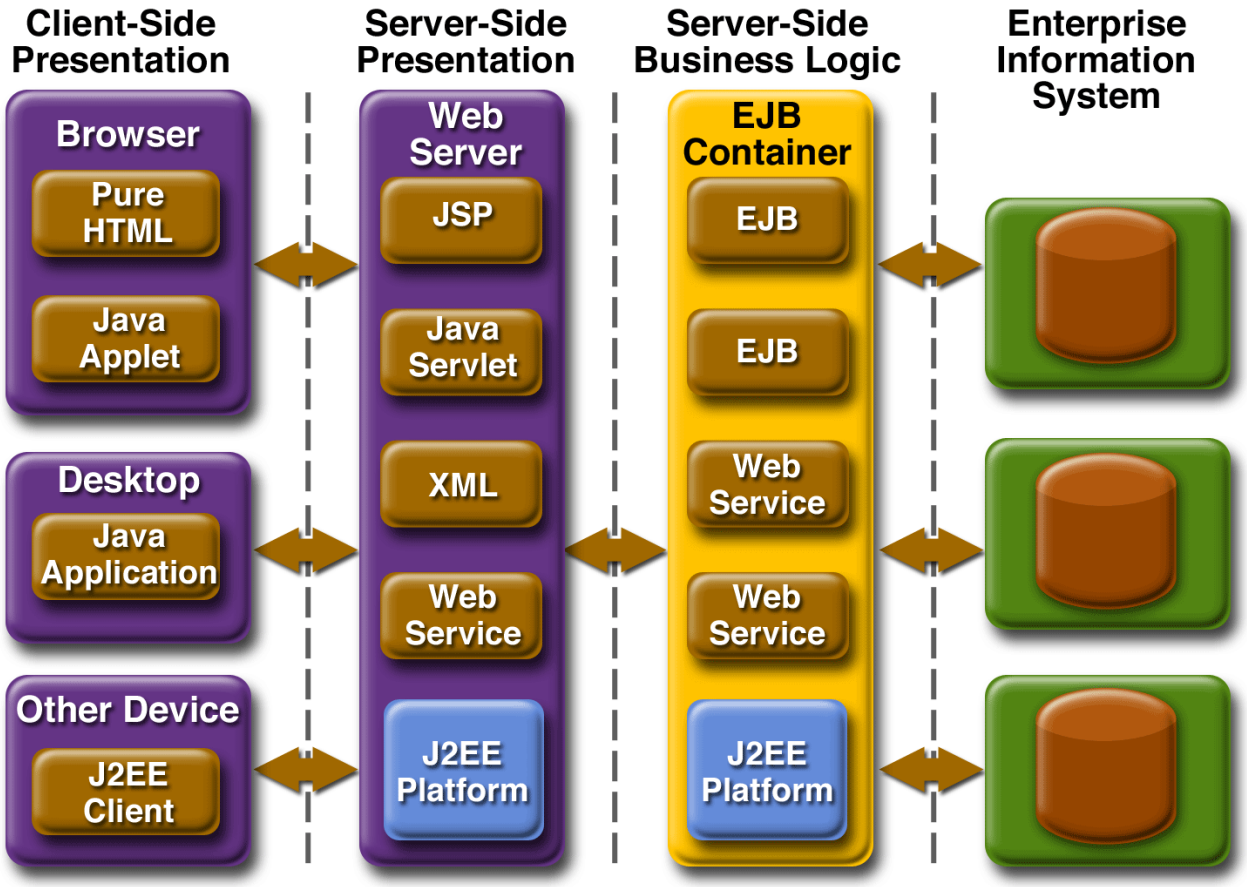
- **JavaEE服务组件用于支持EJB、Servlet、JSP、WebService组件**



JavaEE平台解决方案

- 针对企业解决方案的平台
  - J2SE：可移植（Write Once，Run Anywhere ）、JDBC、 CORBA、安全模型
  - JavaEE支持的企业组件：EJB、Servlet、JSP、XML、WebService
- 更容易的中间件
  - 将复杂的工作交给中间件
    - 快速开发与部署
    - 可移植性与伸缩性
    - 集成遗留资产的困难性
  - 统一的JavaEE标准、统一的基于组件的应用模型
  - 简单性、可移植性、可伸缩性、已有资产集成
- 工业标准

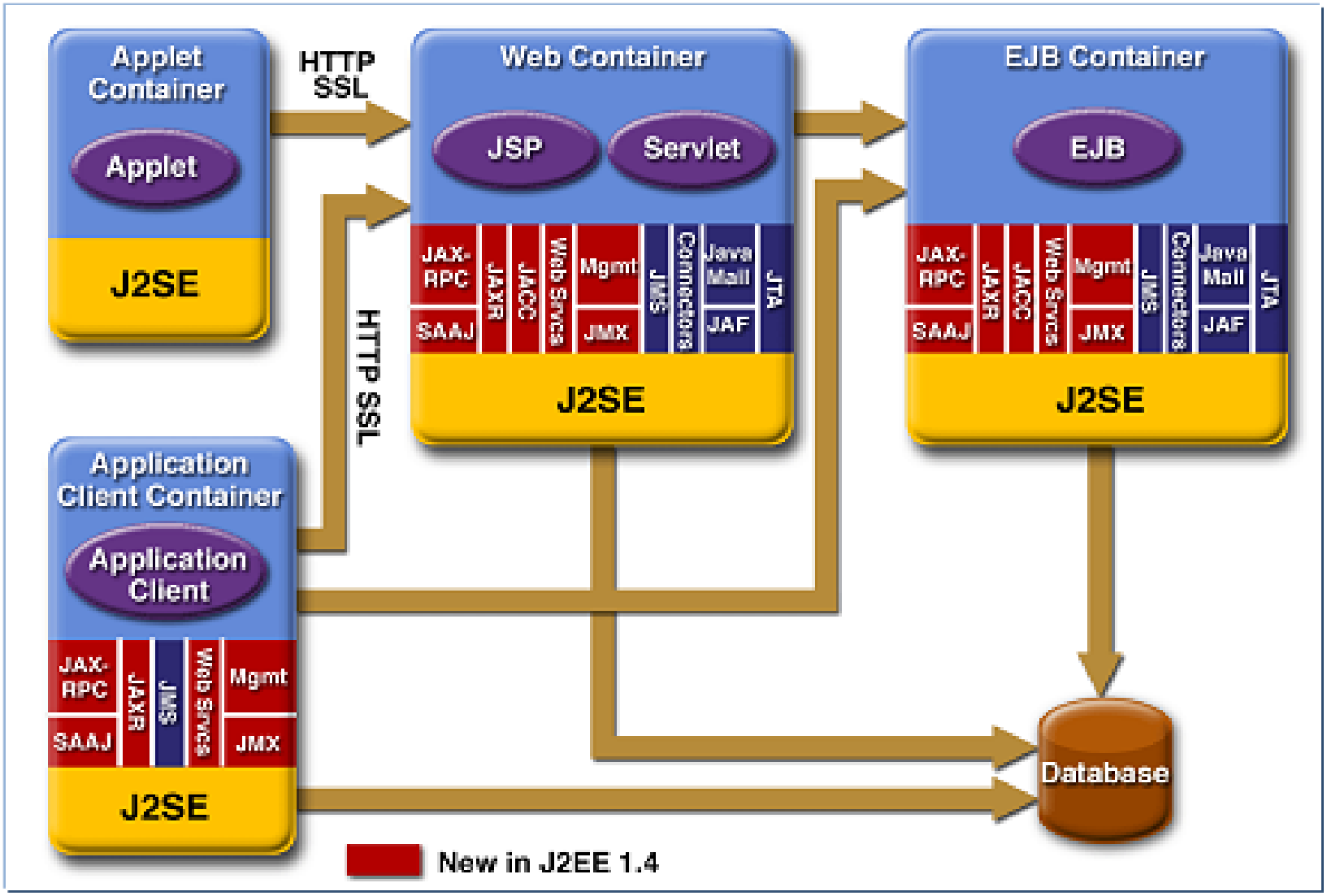
JavaEE蓝图：企业应用模型与最佳实践



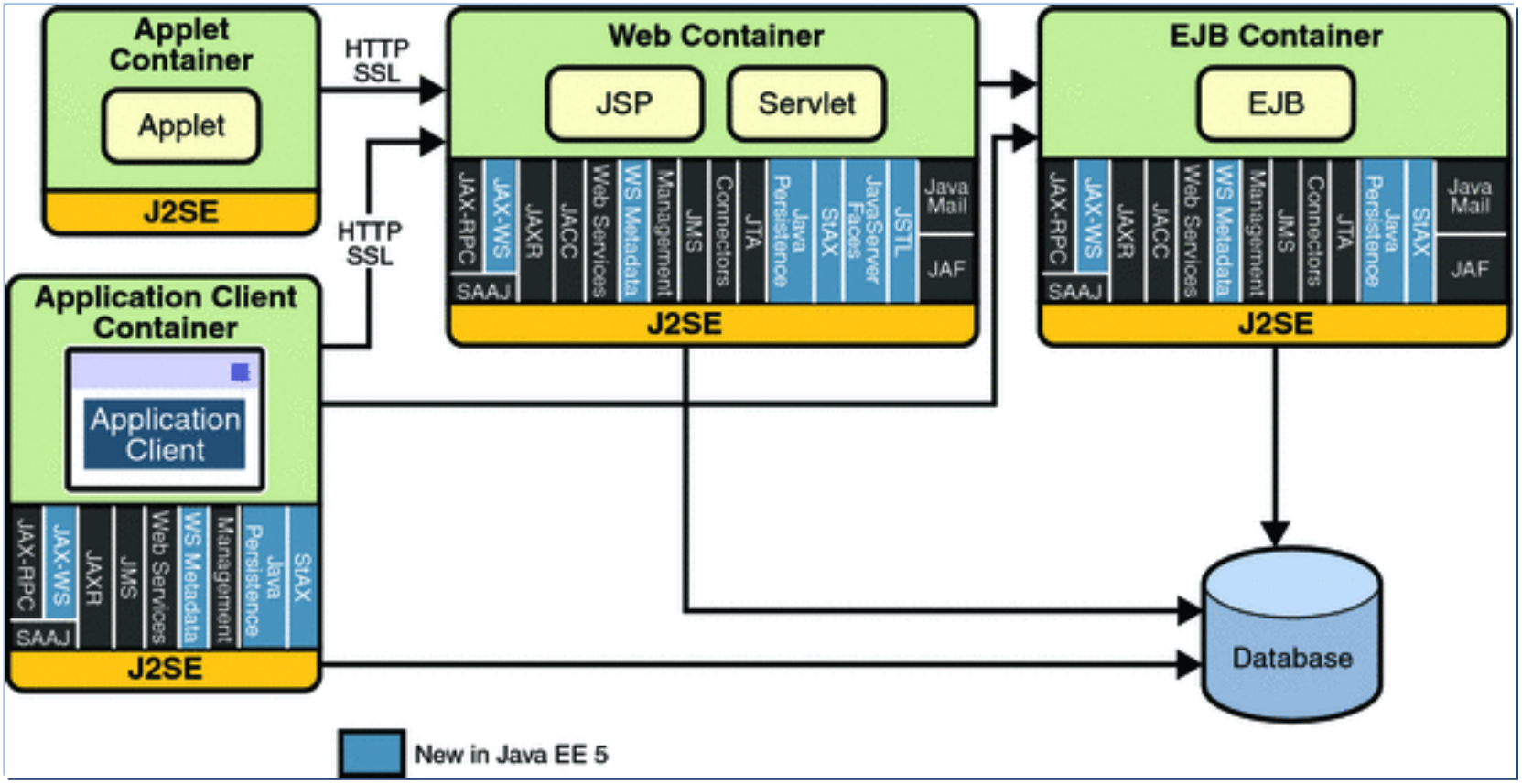
JavaEE的优势

- **容器与连接器：隐藏复杂性，增强可移植性**
  - 组件是应用开发的核心
  - 容器和连接器由中间件提供商实现
- **灵活的用户交互**
  - 桌面、笔记本、PDA、手机、其它设备
  - 独立客户端、HTML、Java applets
  - 服务器端：Servlet、JSP
- **EJB组件模型**
- **与WebService互操作性**
- **加速开发与部署**

JavaEE1.4 Platform APIs



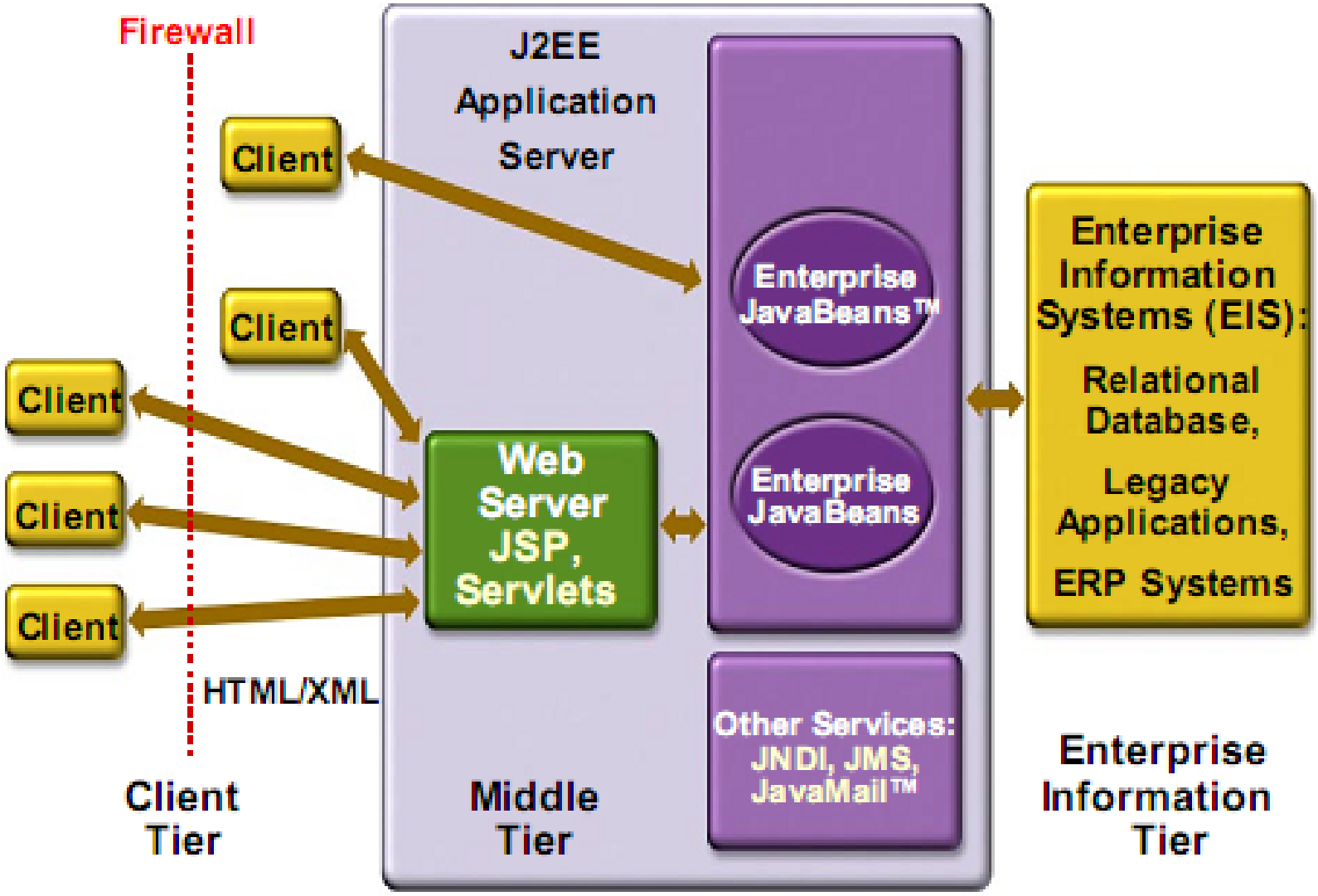
Java EE5 Platform APIs



应用服务器

- JavaEE应用程序要发布在应用服务器中才能运行
- 仅包含Web容器的应用服务器
  - Tomcat、Weblogic Express
- 包含Web容器和EJB容器的应用服务器
  - Weblogic
  - WebSphere

JavaEE应用程序解决方案



基础知识

- JavaEE基础
- Struts框架
- Spring框架
- Hibernate框架
- 浏览器界面技术

Struts框架

- **成熟的Web MVC框架**
- **Struts1.x：简单、高侵入**
  - Servlet控制器
  - Action业务代理
  - ActionForm
  - Struts-config.xml
- **Struts2.x：灵活、低侵入**
  - 基于WebWork的Web MVC框架
  - 灵活的拦截器机制
  - 较低的侵入性

机关保系统使用的Struts框架

- **Struts1.x**
  - 开发简单、培训与学习成本低
  - 有大量的参考资料
  - 独立使用Struts做为Web框架
  - 未使用Spring插件进行集成

基础知识

- JavaEE基础
- Struts框架
- Spring框架
- Hibernate框架
- 浏览器界面技术

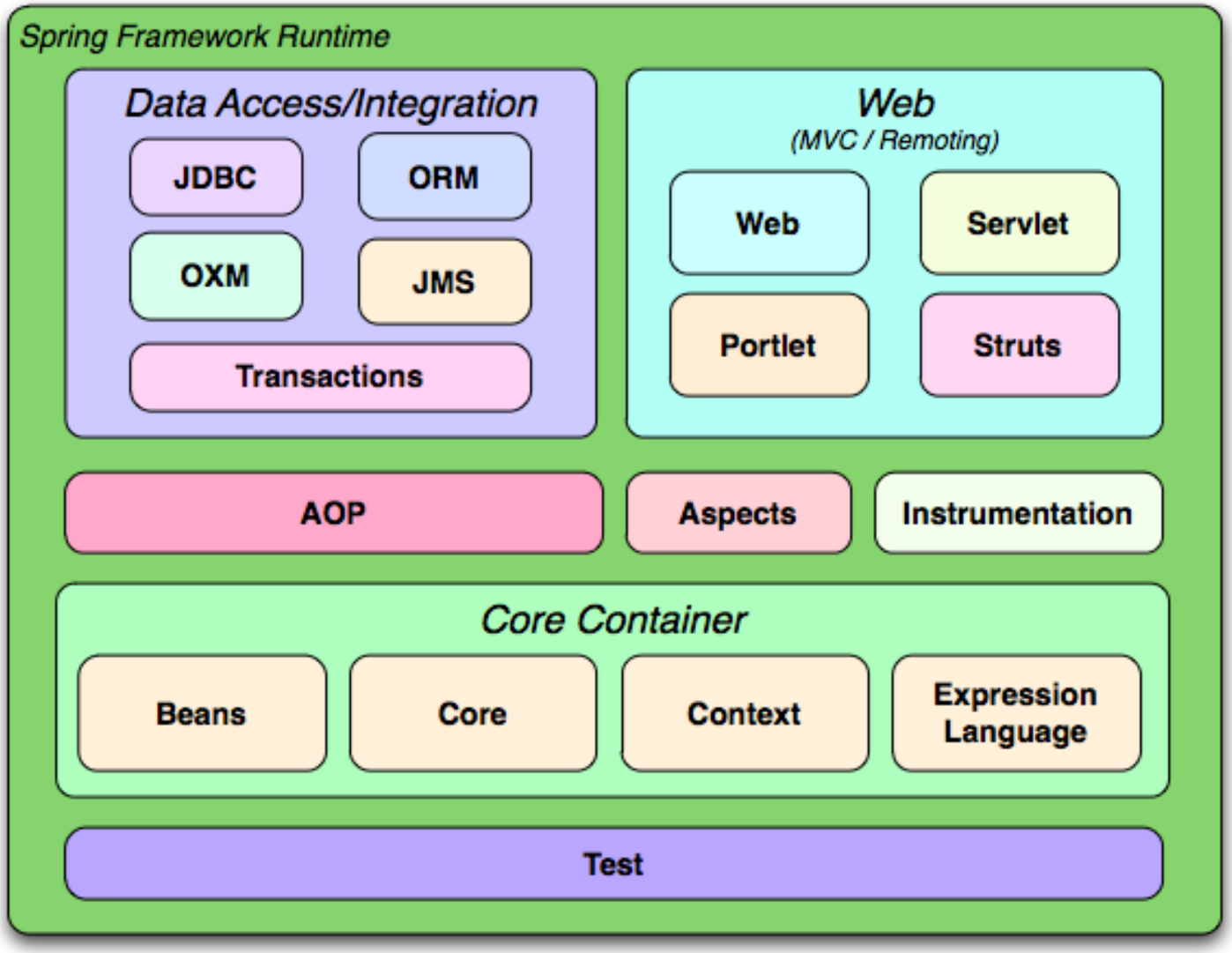
Spring框架

- **什么是Spring**

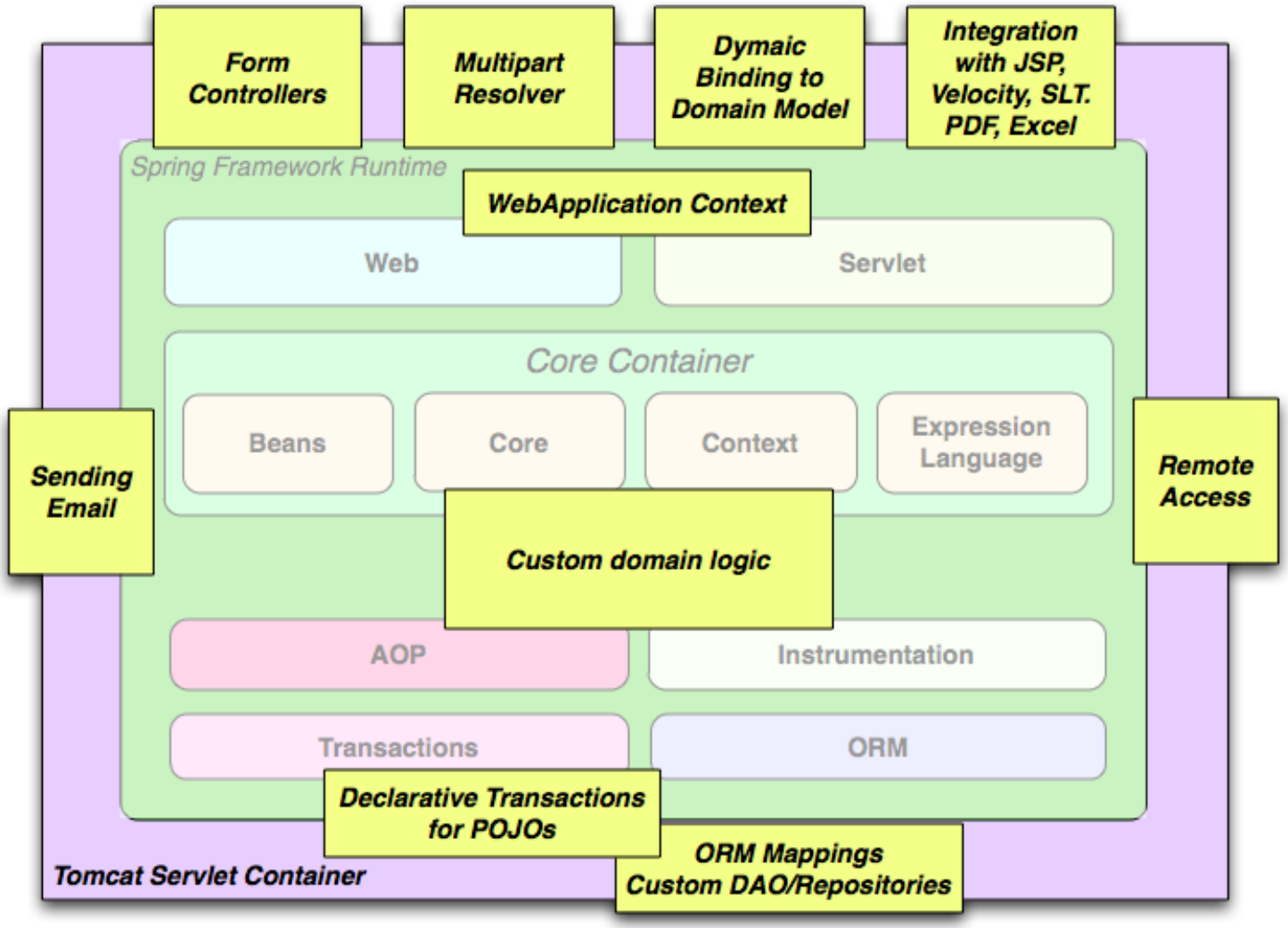
即使有好工具和好技术，开发软件仍然是比较困难的。有一些平台，它们包打天下， 但实际上很沉重、难以控制，在开发过程中效率不高，却让开发软件变得更加困难。 Spring为编写企业应用程序提供了轻量的解决方案，同时仍然支持使用声明式事务、用RMI或web service远程调用、以及使用多种方式来将数据持久化到数据库。Spring提供了全功能的 MVC framework， 以及透明集成AOP到你的软件中的能力。

Spring可能是你的企业应用程序所需要的一站式解决方案， 但Spring仍然是模块化的，允许你只使用你所需的哪些部分，而无需附加上其他部分。 你可以使用 IoC容器，在其上使用Struts，但是你也可以选择使用 Hibernate 整合代码或者 JDBC 抽象层。 我们将Spring设计为非侵入式的（并且以后也是如此），这意味着应用基本上不需要依赖框架本身 （或者肯定是最小的，取决于所使用的部分）。

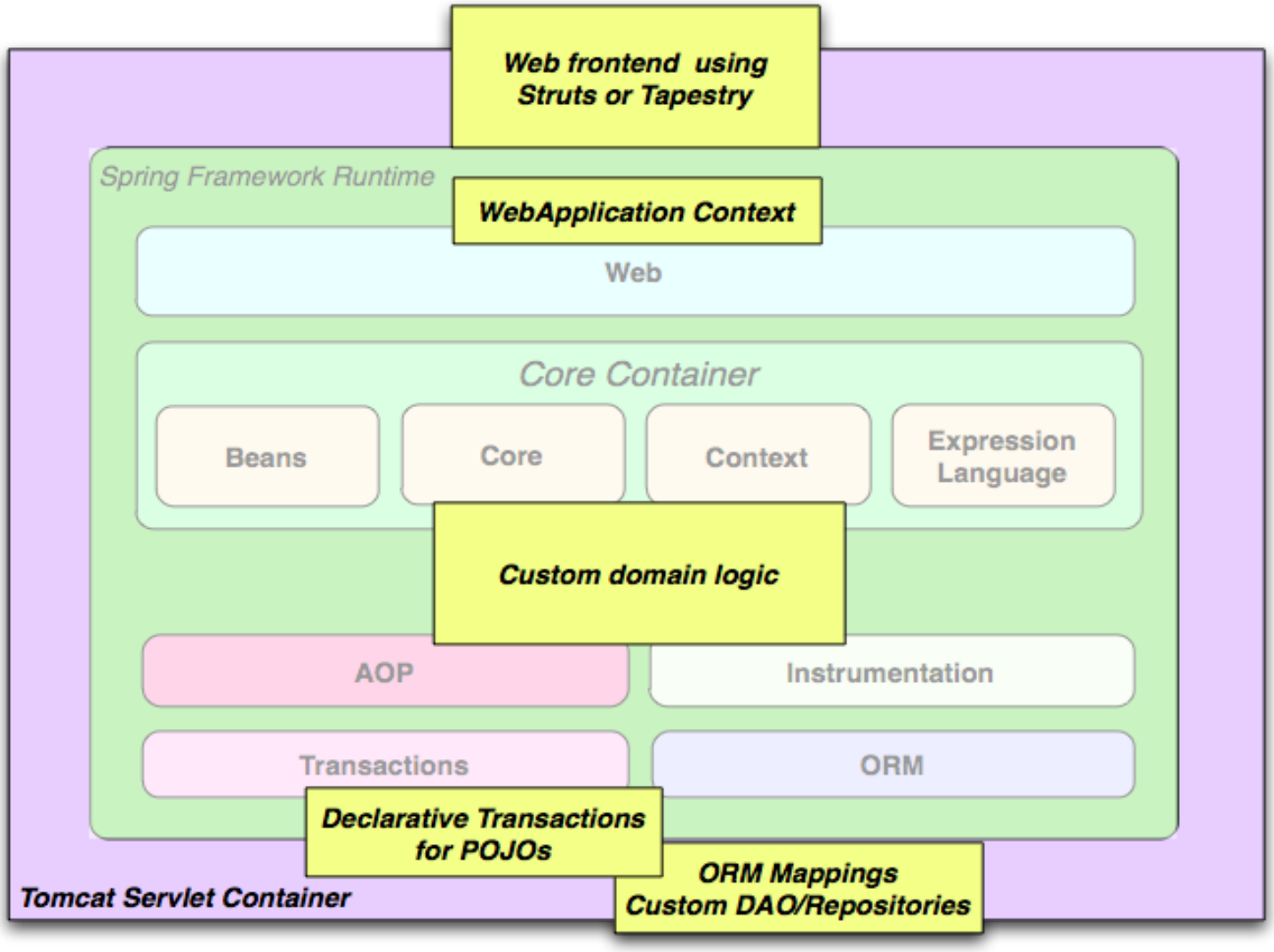
Spring框架Overview



Spring框架完整应用



Spring框架：使用第三方Web框架



Spring框架

- **成熟的中间层框架与容器**
  - 不依赖于应用服务器，在应用服务器之内或之外都能使用
  - 灵活、强大的控制反转（IOC）容器
  - 支持面向切面（AOP）的编程
  - 对中间层数据库访问提供了事务支持、DAO支持、JDBC以及Hibernate等多种ORM工具的集成支持
  - 灵活的Web层MVC框架，可以与Struts等其它Web框架集成
  - 为各种主流技术提供了整合与方便的支持工具：远程访问、Web Service、EJB、JMS、Mail、定时器、动态语言等
- **Spring2.5.x**
  - 兼容Java1.4.2，完全支持Java6
  - 兼容J2EE1.3，完全支持JavaEE5
- **Spring3**
  - 基于Java5，完全支持Java6
  - 兼容J2EE1.4和JavaEE5，引入JavaEE6支持

Spring2.5.x基础：核心技术

- **IOC（控制反转）容器**
  - 通过配置文件或注解方式实现依赖关系的注入，管理应用的各种Java Bean对象（服务对象）
  - 可灵活定义Bean的作用域（单例、每次都创建新对象）
  - 灵活的Bean生命周期回调（可用于完成应用的初始化与销毁工作）
  - Bean定义的继承增强了配置的灵活性
  - 容器扩展点使得容器本身可被灵活定制
  - ApplicationContext增强了国际化、事件、Web应用支持
    - MessageSource、容器事件、ContextLoaderListener 和ContextLoaderServlet
  - 支持基于Java5注解的组件定义和依赖注入配置，在Classpath中进行组件的注入配置扫描


Spring2.5.x基础：核心技术

- 资源
  - Resource接口访问各种资源： UrlResource 、ClassPathResource、 FileSystemResource、ServletContextResource 、 InputStreamResource、ByteArrayResource
  - 将Resource作为属性配置

```
<property name="template" value="some/resource/path/myTemplate.txt"/>
<property name=template" value="classpath:some/resource/path/myTemplate.txt">
<property name="template" value="file:/some/resource/path/myTemplate.txt"/>
```

- 面向切面编程
  - 提供声明式企业服务，特别是为了替代EJB声明式服务。最重要的服务是声明性事务管理。
  - 允许用户实现自定义切面，用AOP来完善OOP的使用。
  - 支持基于Java5注解和基于xml Schema的面向切面编程
  - 充许直接使用AspectJ进行面向切面编程


Spring2.5.x基础：核心技术

- 测试
  - 单元测试
    - 提供了各种Mock对象：JNDI、Servlet API、Portlet API
    - 提供了必要的单元测试支持类：基于反射的工具方法集ReflectionTestUtils、Spring MVC相关的支持类
  - 集成测试
    - 能够无需部署到应用服务器上或连接其它企业架构就实现集成测试
    - 正确配置Spring IoC 容器上下文
    - 使用JDBC或ORM工具的数据访问。可能包括如SQL脚本，Hibernate query，JPA 实体映射等的正确性验证

Spring2.5.x基础：中间层数据访问

- **健壮、灵活、统一的事务管理**
  - 支持JDBC事务和JTA事务，且可通过配置文件进行灵活的切换
  - 声明式事务管理
    - 可在配置文件中灵活声明事务属性：事务传播（ required、RequiresNew）、隔离级别、读写、超时、触发事务回滚的条件（RutimeException）
    - @Transactional注解支持
  - 编程式事务管理
    - 高层次的事务管理API： TransactionTemplate
    - 低层次的事务管理API： PlatformTransactionManager


Spring2.5.x基础：中间层数据访问

- **DAO支持**
  - 一致的异常层次
    - 以DataAccessException 为根，抽象出合理的异常层次
  - 一致的DAO支持抽象类
    - JdbcDaoSupport 、HibernateDaoSupport 、JdoDaoSupport 、JpaDaoSupport
- **JDBC数据访问**
  - 利用JDBC核心类控制JDBC的基本操作和错误处理
    - JdbcTemplate、 NamedParameterJdbcTemplate、SimpleJdbcTemplate
    - SimpleJdbc类：利用JDBC驱动所提供的数据库元数据简化JDBC操作
  - 控制数据库连接
  - 用Java对象来表达JDBC操作：面向对象的方式访问数据库
  - 处理BLOB 和 CLOB对象

Spring2.5.x基础：中间层数据访问

- **Spring提供了对多种ORM工具的集成 支持，使这些ORM工具都能利用Spring一致的异常层次、一致的DAO支持**
- **使用ORM工具进行数据访问**
  - Hibernate
    - 资源管理、 HibernateTemplate、HibernateDaoSupport、编程式事务、声明式事务
  - JDO
  - Oracle TopLink
  - iBATIS SQL Maps
  - JPA


Spring2.5.x基础：Web

- **Spring Web MVC framework Web框架**
  - 支持传统（基于Servlet）的Web开发，围绕DispatcherServlet设计
  - 包括可配置的处理器（handler）映射、视图（view）解析、本地化（local）解析、 主题（theme）解析以及对文件上传的支持
  - 集成多种视图技术： JSP和JSTL 、 Tiles、 Velocity和FreeMarker、 XSLT、文档视图（PDF/Excel）、 JasperReports
- **集成其它Web框架**
  - JavaServer Faces (JSF)、 Struts、 Tapestry、 WebWork
  - 很多Web框架本身都提供了与Spring集成的插件：Struts2
- **Portlet MVC框架**
  - 支持JSR-168 Portlet开发


Spring2.5.x基础：Web

- **集成其它Web框架： Struts1.x**
  - ContextLoaderPlugin：加载Spring配置文件
    - DelegatingRequestProcessor

<action path="/user"/>

    - DelegatingActionProxy

<action path=“/user”
type=“org.springframework.web.struts.DelegatingActionProxy
“ …

<bean name="/user" scope="prototype" autowire="byName"
class="org.example.web.UserAction"/>

  - ActionSupport类集
    - ActionSupport 、DispatchActionSupport、LookupDispatchActionSupport 、MappingDispatchActionSupport

Spring2.5.x基础：整合

- **远程访问与Web服务**
  - 通过RMI基础设施透明的暴露服务
  - 使用Hessian或者Burlap通过HTTP远程调用服务
  - 使用HTTP调用器暴露服务：使用标准Java序列化机制
  - Web Services
    - 使用JAX-RPC暴露web服务
    - 使用JAX-RPC访问web服务
    - 使用JAX-WS暴露web服务
    - 使用JAX-WS访问web服务
  - JMS
- **EJB集成**
  - 访问本地或远程无状态Session Bean，支持EJB2.x和EJB3
  - 提供辅助类实现EJB组件：支持EJB2.x和EJB3
- **JMS**
  - 发送和接收消息，支持同步和异步

Spring2.5.x基础：整合

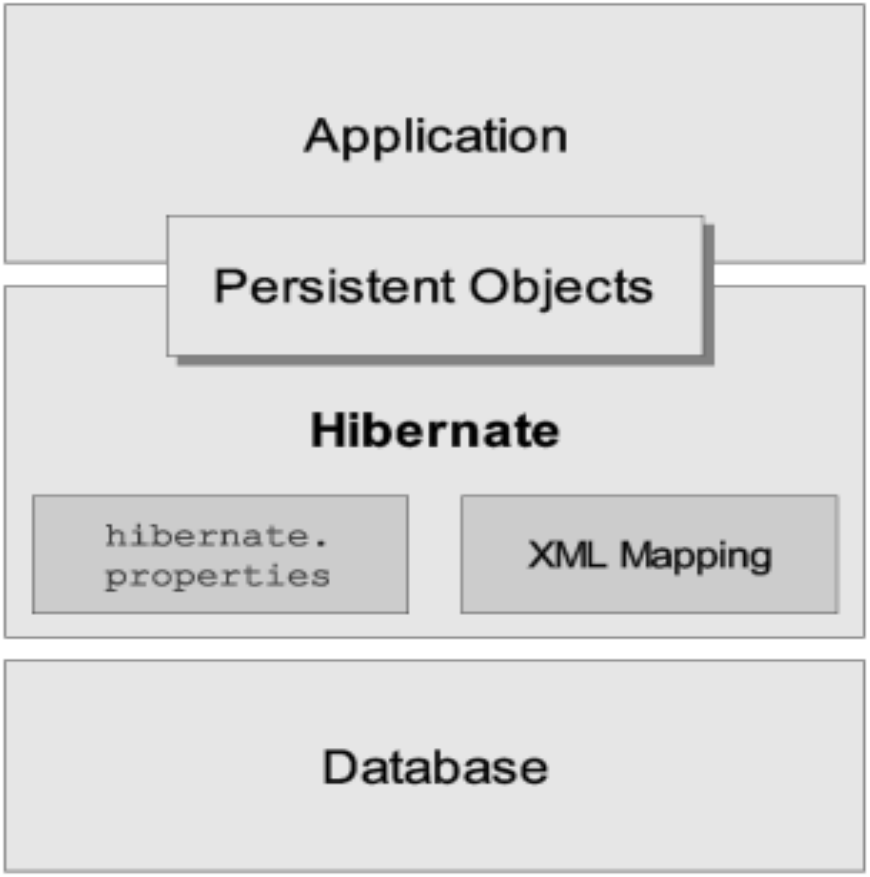
- **JMX**
  - 将 *任意* Spring Bean自动注册为JMX MBean
  - 灵活操纵Bean管理接口的机制
  - 通过远程JSR-160连接器对MBean的声明式暴露
  - 对本地和远程MBean资源的简单代理
- **JCA CCI（Java Connector Architecture Common Client Interface）**
- **邮件抽象层**
- **定时调度(Scheduling)和线程池(Thread Pooling)**
  - 集成Quartz 调度器
  - 提供JDK Timer支持类
- **动态语言支持**
- **注解和源代码级的元数据支持**
  - Java5注解支持、Jakarta Commons Attributes集成

基础知识

- JavaEE基础
- Struts框架
- Spring框架
- Hibernate框架
- 浏览器界面技术

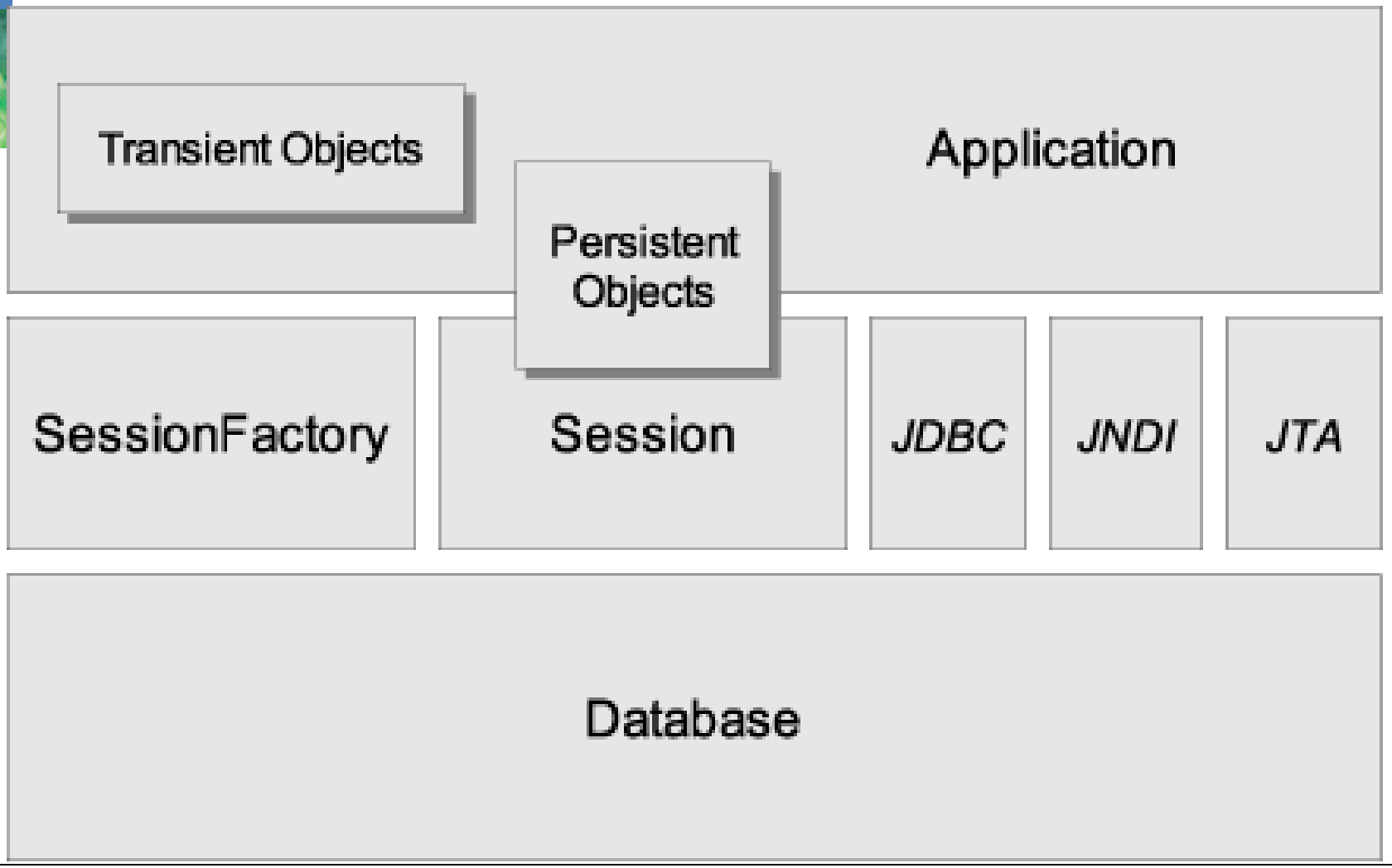
Hibernate框架

- 符合Java习惯的关系数据库持久化（ORM）框架



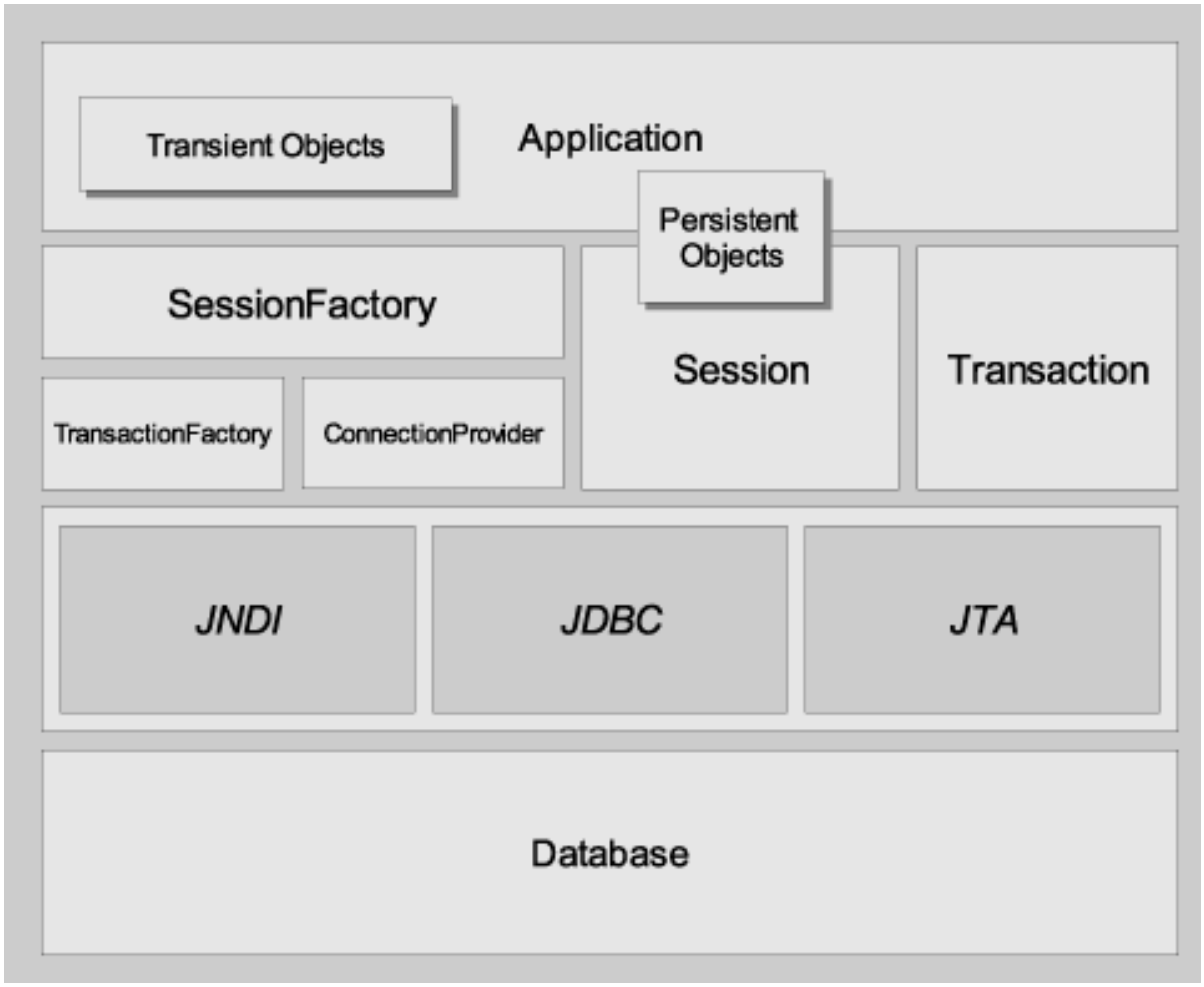
Hibernate框架

- 轻量级解决方案



Hibernate框架

- 全面解决方案



Hibernate框架

- **支持灵活的关联关系映射**
  - 一对一、一对多、多对一、多对多
  - 单向关联、双向关联
  - 继承
- **事务和并发**
  - 提供了非常方便易用的乐观锁支持
  - 支持悲观锁定
- **灵活易用的HQL查询语言**
- **条件查询（Criteria Queries）**
- **Native SQL查询**
- **可以与JDBC一起使用**

基础知识

- JavaEE基础
- Struts框架
- Spring框架
- Hibernate框架
- 浏览器界面技术

浏览器界面技术

- **W3C标准技术**
  - DHTML、Javascript、CSS、Ajax
  - 基于W3C标准形成了很多界面技术框架
    - JQuery、Prototype、Mootools
    - Dojo、YUI、ExtJS、ZK、SmartClient
    - Google Web Toolkit（GWT）
- **浏览器插件技术**
  - 解决HTML标准技术无法支持的功能，如：动画、视频、报表打印、客户端本地设备与资源访问等。
  - Flash、Java Applet、ActiveX、SilverLight

浏览器界面技术的选择

- **Web UI框架**
  - JQuery、Prototype、Mootools
    - JS基础框架，可作为UI组件开发的基础
    - 可轻易获得跨浏览器、简化开发的好处
    - 有很多第三方的扩展组件可以直接使用，但是很多组件需进行修改才能满足业务需要
  - Dojo、YUI、ExtJS、ZK、SmartClient
    - 提供了很多可以开箱即用的UI组件，同时也有很多第三方扩展可用
    - 有非常完整的UI组件库，但也相对比较庞大
  - Google Web Toolkit（GWT）
    - 将Java代码转换为Javascript，可以获得纯JS框架无法获得的好处，如针对浏览器的优化等，是一种新的Web开发思路
    - 很多纯JS框架也同时提供了GWT包装

浏览器界面技术的选择

- **浏览器插件技术**
  - Flash
    - 跨浏览器和操作系统，是目前最成熟、使用最广泛的RIA解决方案
    - 几乎所有的浏览器都已经安装了Flash插件
  - Java Applet
    - 基于Java的RIA解决方案，同样跨浏览器和操作系统，但插件安装较为复杂
  - ActiveX
    - 只能在IE中使用，但是使用非常广
  - SilverLight
    - 微软的RIA解决方案



机关保系统架构概述

系统架构概述

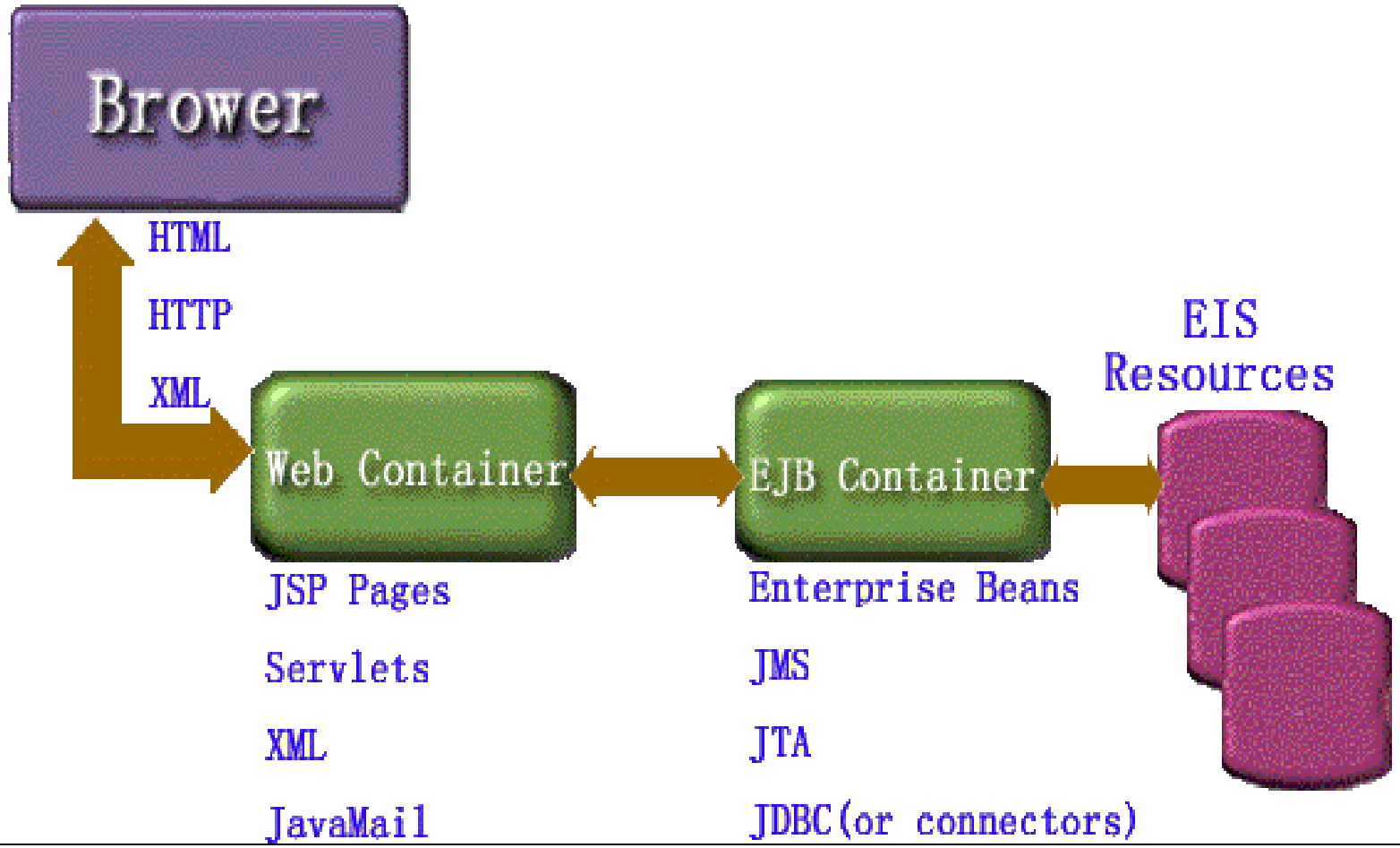
- **架构概览**
- 系统架构对技术问题的解决
- 系统框架介绍
- 与核心平台三版系统架构的比较
- 系统框架的扩展与定制
- 技术标准与环境介绍

架构概览

- JavaEE架构基础
- 对Spring框架的应用
- 对Hibernate框架的集成与应用
- Struts1.x框架的应用
- 界面显示技术
- 机关保系统架构总览
- 机关保系统的分层结构与各层职责
- 系统的数据流模型

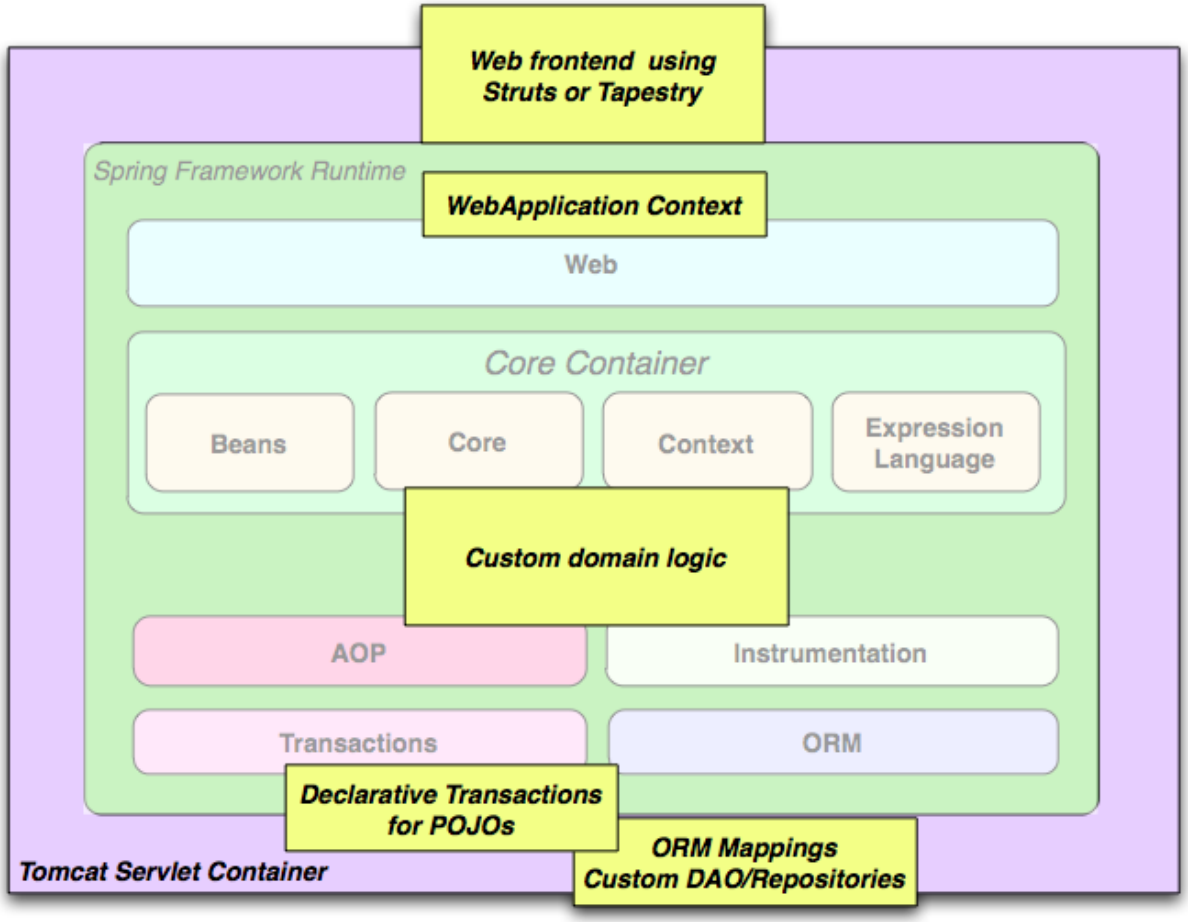

JavaEE架构基础

- 机关保系统的系统架构是基于JavaEE平台的B/S/S多层应用体系结构。



基于Spring的系统架构

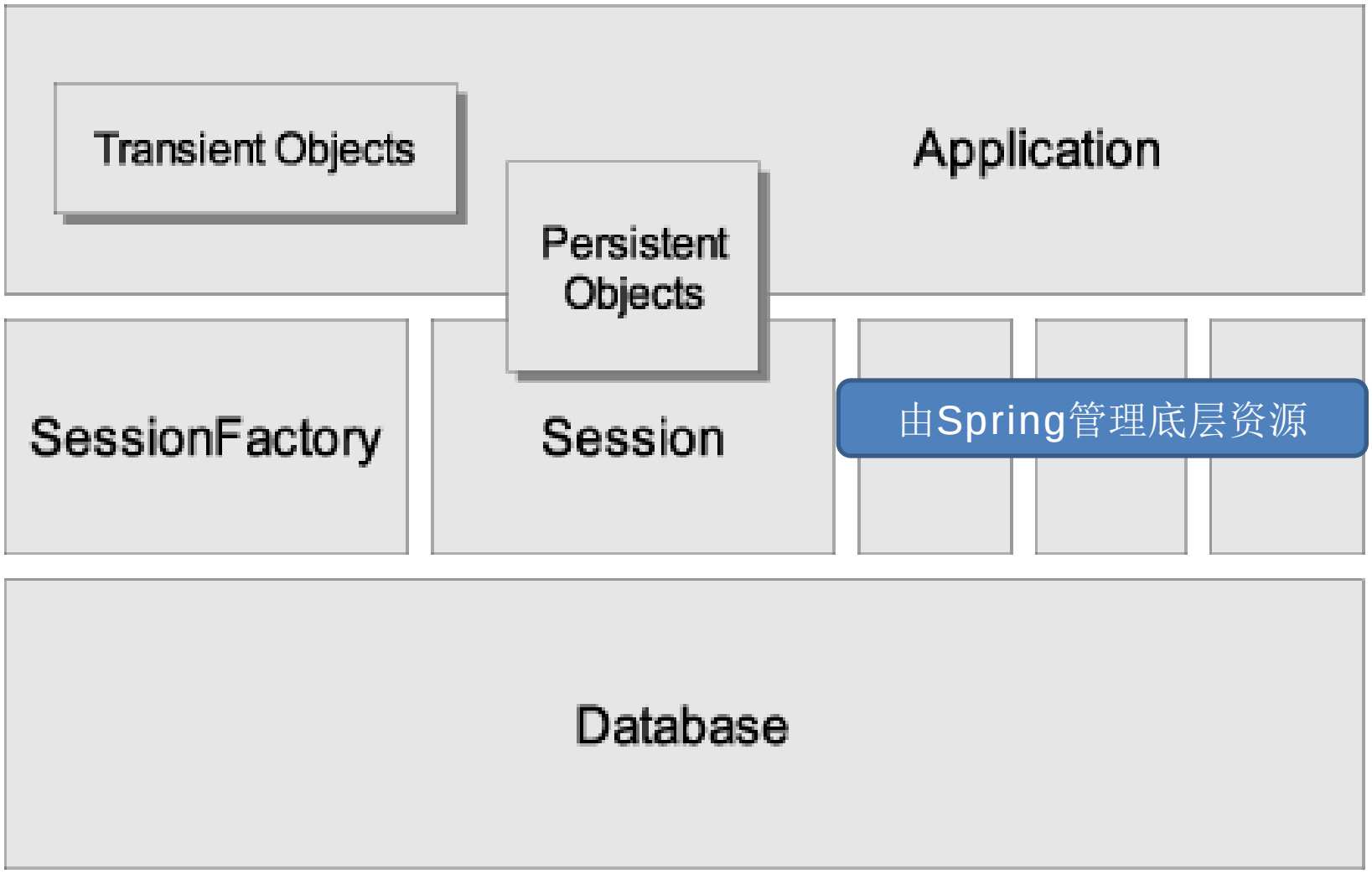
- Spring为企业应用程序提供了一站式解决方案，可以极大地提高开发效率和应用的灵活性



基于Spring的系统架构

- **系统架构整体基于Spring2.5.x**
  - 使用Struts1.x框架做为Web框架，中间层（业务层）采用Spring进行管理，没有使用Spring Web MVC
    - 当前的企业应用中Struts1.x的应用最为广泛，避免了技术切换的学习成本
  - 以Spring为基础的系统架构为系统带来了极大的灵活性和可扩展性
    - 系统框架提供的各种API服务都可以注册为Spring管理的Bean，必要时可以被应用系统扩展或替换为特定于应用的实现
    - 系统的业务组件都开发为Spring管理的Bean，可以通过Spring轻易地为业务Bean添加声明式的事务控制、添加AOP控制
    - Spring IOC容器降低了组件间的藕合
    - 充分利用Spring提供的各种基础设施和便利API使得开发人员更专注于业务罗逻开发：如EJB、WebService、定时计划等

Hibernate ORM框架的应用




Hibernate ORM框架的应用

- **轻量级解决方案**
  - 由Spring容器管理底层的JDBC连接、JNDI、JTA等资源，可以充分利用Spring声明式事务支持等好处，由Spring管理连接也降低了应用管理连接资源的复杂度


Web框架的选择

- **沿用Struts1.x作为Web层框架**
  - 已被广泛使用的Struts1.x框架完全能满足业务需求
  - 避免了转向新框架的学习成本和转换成本
- **Struts1.x与Spring的整合**
  - 独立使用Struts框架，没有采用Spring提供的Struts1.x插件和DelegatingRequestProcessor进行集成
    - 保持了原有的基于Struts框架的开发习惯，降低学习成本
    - 通过在Struts Action中用API获取Spring管理的Bean组件，最大程度上减少了配置文件的变化
    - 不需要在Spring配置文件中对Action类进行配置

界面显示技术

- **使用JavaScript框架支持跨浏览器的快速界面开发**
  - 可任选开源界面组件库或界面开发的商业产品进行界面开发
  - 机关保系统提供了一套JSP标签库作为参考实现
- **其它辅助界面技术**
  - 根据需要可选用Flash技术实现图表、打印、界面特效等功能。甚至可以完全使用Flex技术实现所有界面。
  - 使用Java Applet插件技术进行无需预览的Jasperreports报表打印控制功能

机关保系统架构概览

业务系统

授权管理 | 日志管理 | 系统管理 | 系统参数 | 通用审批

缓存服务 | 通用回退服务 | 服务管理API | 编码过滤器

日志服务 | 参数配置服务 | Excel读写服务 | 日志过滤器

JSON服务 | 查询辅助API | 报表打印服务 | 安全过滤器

持久化服务 | 分页查询服务 | 树构造API | 浏览器缓存过滤器

数据校验服务 | 代码表缓存服务 | 。。。

系统框架

按钮 | 下拉列表 | 表格

输入框 | 日期时间 | 工具栏

图表 。。。 | 表单录入标签 | 树

面板

其它标签

UI标签库

Struts1.x框架

Spring框架

Hibernate框架

系统的分层结构

界面组件（Html、JS、CSS、JSP）

请求处理层Action

服务接口层BS

服务实现层BSImpl

面向对象的业务实现层BPO

数据访问层DAO

数据库

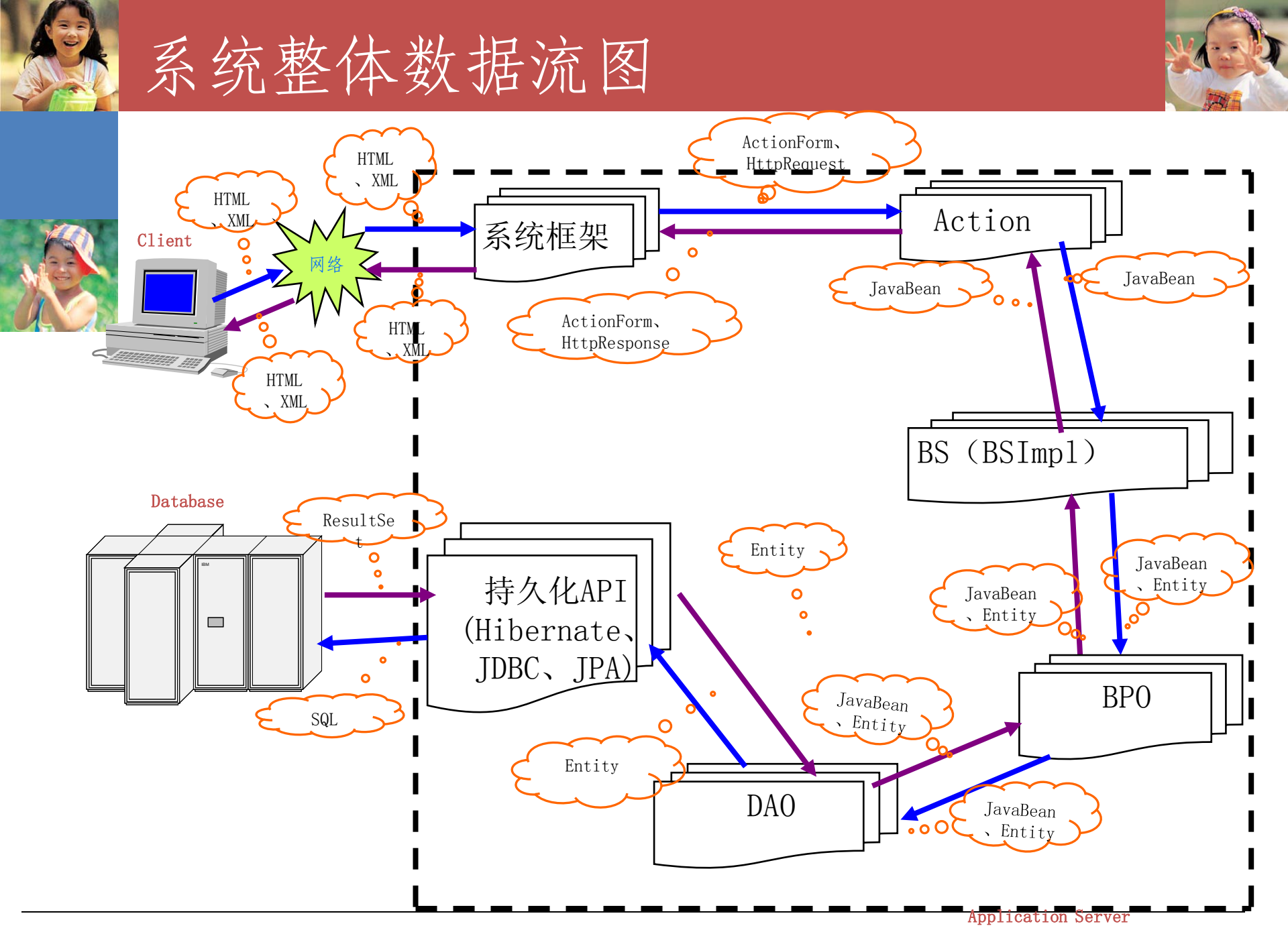
系统各层次的职责划分

- **界面组件（Html、JS、CSS、JSP）**
  - 界面内容显示，用户交互
- **请求处理层Action**
  - 业务代理，校验用户输入的合法性，将请求参数转换为业务数据对象并转交给业务组件进行业务处理，将业务组件返回的业务数据返回给用户界面
  - 捕获并处理异常
  - 基于Struts1.x的Action层必须是线程安全的
- **服务接口层BS**
  - 定义业务组件的接口，面向接口编程，业务接口使业务组件不依赖于具体的实现技术（POJO、EJB、WebService）
- **服务实现层BSImpl**
  - 业务组件的实现，可以通过调用下一层的业务对象实现具体的业务逻辑。
  - 通用配置或注解进行声明式事务处理，或直接使用事务API进行事务处理
  - 可灵活地选择POJO、EJB、WebService等具体技术进行实现
  - BSImpl层应当是线程安全的

系统各层次的职责划分

- **面向对象的业务实现层BPO**
  - BSImpl层的辅助实现层，按照面向对象的方式实现复杂的业务逻辑与算法。对于业务逻辑简单的组件可以省略该层，该层也可以看作是BSImpl层的内部实现。
- **数据访问层DAO**
  - DAO层用于实现复杂的数据库访问逻辑，其主要目的是隔离特定于数据库的数据访问逻辑。通过为不同的数据库产品开发不同的DAO层实现，达到数据库产品迁移的目标。对于简单的业务和没有多数据库支持需求的应用，可以省略该层。
  - DAO层也应该是线程安全的
  - DAO层也可以类似VS和VSImpl层一样抽象出一个接口层，以实现面向接口编程和更好的扩展性。

系统整体数据流图


HTML
、XML
HTML
、XML
Client
网络
系统框架
ActionForm、
HttpRequest
Action
JavaBean
JavaBean
HTML
、XML
ActionForm、
HttpResponse
HTML
、XML
BS（BSImpl）
Database
ResultSe
t
持久化API
(Hibernate、
JDBC、JPA)
Entity
JavaBean
、Entity
JavaBean
、Entity
SQL
BPO
JavaBean
、Entity
Entity
DAO
JavaBean
、Entity
Application Server

系统架构概述

- 架构概览
- 系统架构对技术问题的解决
- 系统框架介绍
- 系统框架的扩展与定制
- 技术标准与环境介绍

系统架构对技术问题的解决

- 业务开发模型与核心API
- 各种技术问题的解决方案
  - 通用回退
  - 通用审批
  - 数据源管理
  - AOP支持与事务管理
  - Bean（服务）管理
  - 数据访问方式
  - 远程访问与Web服务支持
  - EJB支持
  - JMS、邮件等支持
  - 定时任务支持
  - 注解支持
  - 报表支持
  - LOB字段支持
  - 缓存

系统架构对技术问题的解决

- 业务开发模型与核心API
  - UI界面层
    - 使用标签库简化开发，辅助使用JS解决标签库行为定制和特殊交互的问题；
    - 必要时使用Flash、Java插件等技术实现特殊的界面需求（如报表打印、统计图表等）
  - Action层
    - 框架提供getService api获取Spring管理的业务Bean服务对象
    - 框架提供分页查询API，与UI标签库相配合实现灵活的通用分页查询、导出Excel功能
    - 框架提供返回Json等格式数据的API用于响应Ajax请求

系统架构对技术问题的解决

业务开发模型与核心API

- **BS与BSImpl层**
  - 框架提供基于**AOP**的事务支持，可以通过在配置文件中配置事务规则或使用**Java5**注解指定事务属性
  - 可以通过**Spring**依赖注入或框架**AppContext**类中的**API**获取系统服务组件（如**Hibernate**持久化**API**、参数配置对象、缓存等）
- **BPO层**
  - 可引用的业务组件
  - 可借助**Spring**提供的基础设施将**BSImpl**实现为**EJB**、**WebService**等组件
- **DAO层**
  - 通过依赖注入或**AppContext**类的**API**获取系统服务组件（主要是**Hibernate**或**JDBC**持久化组件）
- 应用生命周期管理与通用过滤器
  - 框架提供的ApplicationInitListener类管理应用的初始化和关闭
  - 系统提供中文编码处理、缓存处理、日志处理、权限过滤器等组件进行相关方面的通用处理

通用保存前台代码

```
var tbar = [{
        text : '保存[S]',
        id:'corReg',
        icon : jsapp.ctxPath+'/res/images/icons/standard/save.gif',
        handler : function(){
            //调用架构通用提交方法
            executeSave(
                function(result){
                submitForm('form2', url,function(res){
                    if(res.success == true){
                        Ext.getCmp('corReg').setDisabled(true);
                        Ext.getCmp('archiveBtn').setDisabled(false);
                        }
                });

            });
        }
```

通用保存后台代码

```
/**
 * XXXXXBS实现类
 */
public class CorporInfoBSImpl extends ServiceBase implements CorporInfoBS{

    /**
     * XXX保存
     */
    public void saveEffectInsuInfo(CorporInfoDTO dto) {

        //处理业务日志
        Long aaz002 = getOrNewAaz002(dto);
        //保存业务日志ID
        dto.setAaz002(aaz002);
        //保存界面还原
        saveHtml(dto);
        //业务逻辑处理

        //通用复核方法调用
        saveFlow(dto);
    }
```


系统架构对技术问题的解决

- 业务开发模型与核心API
  - UI界面代码模板

```
<%@ page contentType="text/html; charset=UTF-8"%>
<%@ taglib uri="http://brick.bjlbs.com.cn/tags-spot" prefix="s"%>
<html>
<s:head title="标题">
<script type="text/javascript">
//自定义JS代码
</script>
</s:head>
<body>
<s:onready>
    <!-- JSP标签-->
</s:onready>
</body>
</html>
```

系统架构对技术问题的解决

- 业务开发模型与核心API
  - Action层代码模板

```
public class XXBizAction extends BizAction {
XXBizBS bs = getService(XXBizBS.class);

public ActionForward addPage(final ActionMapping mapping,
final ActionForm actionForm, final HttpServletRequest request,
final HttpServletResponse response) throws Exception {
// ....
bs.biz();
// ....
return mapping.findForward("add");
}
}
```

系统架构对技术问题的解决

- 业务开发模型与核心API
- BS层代码模板

```
public interface XXBizBS {

void biz();

void biz2();

}
```


系统架构对技术问题的解决

- **业务开发模型与核心API**
  - **BSImpl层JDK1.4风格的代码模板**

```
/**
 * JDK1.4风格的代码模板。可从父类继承jdbc、hibernate、config对象，其它属性
     可以自动注入（必须有getter、setter方法）或使用API获取。
 */
public class Jdk14BizVSImp extends ServiceBase implements Jdk14BizVS {
// 自动通过setter方法注入已存在的服务
TransactionTemplateEx trans;
// 通过API获取已存在的服务对象
AppConfig appConfig = AppContext.getService(AppConfig.class);
// 通过API获取服务对象，如果服务对象不存在，则自动在Classpath中搜索并自动
     注册服务
XXBizDao dao = AppContext.getService(XXBizDao.class);
public TransactionTemplateEx getTrans() {return this.trans;}
public void setTrans(final TransactionTemplateEx trans) {this.trans = trans;}
```


系统架构对技术问题的解决

- **业务开发模型与核心API**
  - **BSImpl层JDK1.5风格的代码模板**

```
/** JDK1.5风格的代码模板，使用注解注入和声明服务组件 */
@Service("bizBS")
public class Jdk15BizVSImp implements Jdk15BizVS {
@Autowired
JdbcTemplateEx jdbc;
@Autowired
HibernateTemplateEx hibernate;
@Autowired
TransactionTemplateEx trans;
@Autowired
SysConfig config;
@Autowired
Jdk15BizDao dao;
@Transactional(propagation = Propagation.REQUIRES_NEW, readOnly =
    false, rollbackFor = Throwable.class, timeout = 30)
public void biz() {
```

系统架构对技术问题的解决

- 业务开发模型与核心API
  - DAO层JDK1.4风格的代码模板

```
/**JDK1.4风格的代码模板。属性可以自动注入（必须有getter、setter方法）或使用API获取。 */
public class Jdk14BizDao {
JdbcTemplateEx jdbc;
HibernateTemplateEx hibernate;
TransactionTemplateEx trans;
SysConfig config;
// 需要注入的各属性都必须有getter和setter方法
```


系统架构对技术问题的解决

- 业务开发模型与核心API
  - DAO层JDK1.5风格的代码模板

```
/** JDK1.5风格的代码模板，使用注解注入和声明服务组件 */
@Repository
public class Jdk15BizDao {
@Autowired
JdbcTemplateEx jdbc;
@Autowired
HibernateTemplateEx hibernate;
@Autowired
TransactionTemplateEx trans;
@Autowired
SysConfig config;
// 注入的属性不需要getter和setter方法
```

业务流程配置

流程配置SYSBUSINESS表，实现通用回退、表单打印功能配置：

属性名称	功能说明
allowrollback	是否支持通用回退
allowprint	是否支持表单打印
allowcross	是否支持交叉授权
allowsendsms	是否支持短信发送

系统架构对技术问题的解决

- **各种技术问题的解决方案**
  - 通用回退
    - 沿用核三的通用回退方案
  - 通用审批
    - 使用界面还原技术
    - 反射机制
    - 工作流
  - 数据源管理
    - 通过Spring配置DataSource管理数据源，可以通过配置灵活地使用JDBC或JNDI数据源
    - 可以通过配置方便地切换DHCP、C3P0、应用服务器连接池等数据库连接池
  - AOP支持与事务管理
    - 基于Spring提供的AOP支持，可以很容易地支持AOP编程
    - 事务管理是Spring AOP支持实现的一个功能，可以使用配置文件或Java注解指定受管Bean的事务属性
    - 除了使用基于AOP的事务管理之外，系统民提供了事务管理的API

系统架构对技术问题的解决

- **技术问题的解决方案**
  - **Bean（服务）管理**
    - 一般情况下服务组件的作用域都为单例，这能获得良好的性能，且能满足绝大多数的需求
  - 数据访问方式
    - 系统框架同时提供了**Hibernate**和**JDBC**两种**API**服务进行数据库访问
    - 通常情况下只需要使用**Hibernate**进行数据访问就足够了
    - 在需要访问存储过程、特殊的批量数据更新的情况下，系统框架也提供了**JDBC**服务以允许使用**JDBC**访问数据库
    - 通常一个业务要么使用**Hibernate**、要么使用**JDBC**进行数据库访问，一般禁止在同一笔业务中混合使用**Hibernate**和**JDBC**，以避免不必要的复杂化和错误
    - 如果一定要混合使用**Hibernate**和**JDBC**，则一定要万分小心**Hibernate Session**缓存与数据库的同步问题，防止访问到旧的错误数据或写入错误的数据


系统架构对技术问题的解决

- **技术问题的解决方案**
  - 远程访问与Web服务支持、 EJB支持、JMS、邮件等支持
    - 通过Spring提供的基础设施，可以方便地实现或访问这些功能
    - 电子档案集成 、短信平台集成
    - 通过webservice实现网上系统与中心系统对接
  - 定时任务支持
    - 可以使用Timer实现简单的定时任务
    - 使用QuartZ可以实现集群环境中的定时任务
  - 注解支持
    - 可以使用注解代替大部分的配置文件（不是全部），可以使用注解声明Bean、注入Bean、声明事务属性等
  - 报表支持
    - 系统提供了JasperReport报表打印服务
    - 系统提供了一个Java插件用于实现浏览器端的无预览直接打印支持
    - 支持PDF、Word、Excel、HTML等多种格式输出
    - 灵活的报表设计，支持图表

系统架构对技术问题的解决

- **技术问题的解决方案**
  - **Bean（服务）管理**
    - **系统框架提供的服务API都注册为Bean服务，业务组件可以通过注入或API获取和使用框架服务**
    - **系统框架提供了业务组件的自动发现与注册机制，使得业务组件在即使没有被配置或注解为服务时，AppContext.getService API仍然能够自动从Classpath中搜索服务的实现并自动注册为Bean服务，方便应用**
  - **LOB字段支持**
    - **Spring使得不同数据库的Blob、Clob字段处理统一而简单**
  - **缓存**
    - **系统框架提供了一个支持集群的缓存服务API**
    - **系统实现了基于缓存服务的代码表缓存**

系统架构概述

- 架构概览
- 系统架构对技术问题的解决
- 系统框架介绍
- 与核心平台三版系统架构的比较
- 系统框架的扩展与定制
- 技术标准与环境介绍

系统框架介绍

- 框架总览
- 框架的核心API
- 框架服务与控制组件介绍
- UI界面组件

机关保系统架构概览

业务系统

授权管理 | 日志管理 | 系统管理 | 系统参数 | 通用审批

缓存服务 | 通用回退服务 | 服务管理API | 编码过滤器

日志服务 | 参数配置服务 | Excel读写服务 | 日志过滤器

JSON服务 | 查询辅助API | 报表打印服务 | 安全过滤器

持久化服务 | 分页查询服务 | 树构造API | 浏览器缓存过滤器

数据校验服务 | 代码表缓存服务 | 。。。

系统框架

按钮 | 下拉列表 | 表格

输入框 | 日期时间 | 工具栏

图表 。。。 | 表单录入标签 | 树

面板

其它标签

UI标签库

Struts1.x框架

Spring框架

Hibernate框架

框架的核心API

- **框架的核心API**
  - **AppContext**
    - **Bean服务管理的核心API类，提供获取并自动查找服务的API，同时提供了获取常用的系统服务的API（参数配置、缓存服务、代码表缓存服务、Hibernate服务、JDBC服务、事务API服务）**
  - **AppConfig**
    - 应用的参数配置对象，可被应用扩展
  - CodeListManager
    - 代码表缓存服务，用于管理和获取代码表缓存
  - HibernateTemplateEx
    - 扩展了**Spring HibernateTemplate的Hibernate API服务**
  - JdbcTemplateEx
    - 扩展了**Spring JDBCTemplate类JDBC API服务**
  - **BigExcelUtil**
    - 大数据量**Excel读写API**

框架的核心API

- **框架服务与控制组件介绍**
  - 服务
    - 各种服务**API**
    - 大多数服务**API**都可以配置为**Spring Bean**组件，因此可以容易地扩展 或替换这些服务组件
  - 控制组件
    - 应用生命周期管理：初始化、关闭
    - 日志、安全、编码、浏览器缓存过滤器
      - 对业务异常、请求信息进行日志记录
      - 过滤用户权限
      - 对请求参数中的中文信息进行正确的编码处理以防止乱码
      - 防止浏览器对响应数据进行不正确的缓存导致界面刷新错误

UI界面组件

- **UI界面组件**
  - **UI标签库**
    - **以第三方JS界面组件库为基础，可直接使用JS与JSP标签结合进行界面开发**
    - **对JS UI组件库的JSP Tag封装**
    - **通过封装标签简化JSP页面开发**
    - **兼容IE、Firefox多种浏览器**
  - **地图界面组件**
    - **使用开源的OpenLayers作为地图界面组件的实现，不仅支持静态图片作为地图，也支持多种标准的GIS地图接口产生的地图数据**
    - **纯JS方式使用，可根据实际需要进行JSP标签封装或封装为JSP标记使用。**

系统架构概述

- 架构概览
- 系统架构对技术问题的解决方式
- 系统框架介绍
- 系统框架的扩展与定制
- 技术标准与环境介绍

系统架构概述

- 架构概览
- 系统架构对技术问题的解决方式
- 系统框架介绍
- 系统框架的扩展与定制
- 技术标准与环境介绍

系统框架的扩展与定制

- **参数配置**
  - 扩展**AppConfig**或其子类并在**Spring**配置文件件中进行配置
- **持久化API**
  - 扩展**HibernateTemplateEx**和**JDBCTemplateEx**并进行配置
  - 使用或扩展**Spring**提供的其它持久化工具**API**
- **缓存服务与代码表缓存**
  - 扩展ICacheManager定制缓存服务
  - 扩展CodeCacheLoader定制代码表缓存的内容
- **应用的生命周期管理**
  - 使用**JavaEE**标准的**Listener**接口
- **安全过滤器**
  - 扩展**SafetyFilter**
- **Apache BeanUtils自定义类型转换规则**
  - 配置**BeanUtilsConfiger**并指定自定义的类型转换器
- **事务API**
  - 添加新的**TransactionTemplateEx**组件或对其进行扩展

系统架构概述

- 架构概览
- 系统架构对技术问题的解决方式
- 系统框架介绍
- 系统框架的扩展与定制
- 技术标准与环境介绍

系统技术标准

- JavaEE1.4
  - Servlet2.4、JSP2.0、EJB2.1
- JavaEE5
  - Servlet2.5、EJB3.0、JSF1.2、JSTL1.2

环境介绍

- 支持的应用服务器
  - 原则上支持一切遵循JavaEE标准的应用服务器
  - 已测试的应用服务器：Tomcat、Weblogic、TongWeb
- 最终软件运行平台
  - 客户端：Windows
  - 服务器端：几种版本的Unix（Sun Solaris,HP/UX,AIX,DEC Unix）
- 使用的开发平台
  - Eclipse 3.5以上
  - ORACLE 10G

环境介绍

- 支持的应用服务器
  - 支持JavaEE1.4的应用服务器
    - 可获得JavaEE1.4新特性带来的好处（JSP标记、JSP2.0等）
  - 支持JavaEE5的应用服务器
    - 可获取JavaEE1.4新特性带来的好处（JSP标记、JSP2.0等）
    - 同时可使用EJB3.0，大大简化EJB开发

环境介绍

- 几种主流应用服务器对JavaEE标准的支持

JavaEE版本	Tomcat （不支持EJB技术）	Weblogic	Websphere
JDK1.2－1.4 JavaEE1.3	4.1 Servlet2.3、JSP1.2	7.0	
JDK1.4 JavaEE1.3	4.1	8.1	5.1
JDK1.4 JavaEE1.4	5.0 （Servlet2.4、JSP2.0）		6.0
JDK1.5 JavaEE1.4	5.5	9	6.1
Java5 JEE5	6.0	10	7.0

Thank You !