

# 温州医学院

## 本 科 生 毕 业 论 文 （设 计）

题 目 基于 AJAX 技术的护理实验室平台设计与实现

导师姓名 叶夏 导师签字

专业技术职称

学生姓名 陈 达

学院（部） 信息与工程学院

专业班级 05 信管（1）班

完成时间 2009 年 5 月 7 日

温州医学院教务处

# 《基于 AJAX 技术的护理实验室平台设计与实现》

## 毕业论文

作者：陈达 指导老师：叶夏

**【摘要】**该护理实验平台是基于B / S模式应用系统，针对我校护理实验室平台信息容量小，交互性差等缺点专门开发。系统将成熟的开发模式MVC下的Struts2和Hibernate、Spring与Ajax技术相结合，将传统B / S结构中的“哑”客户端改造成富客户端，构建异步通信，实现无刷新web应用，减少HTTP请求中出现的等待现象，提高Web应用程序执行效率。运用ExtJs强大的UI组件改善用户体验，以构建更高效更合理的Web应用框架。

**【关键词】**对象关系映射；依赖注入；面向切面编程；Ext JS； Ajax； AOP编程；ORM；

**【Abstract】** The experimental platform is based on care B / S mode applications, care for our school laboratory platform information capacity of small, specialized interactive poor development shortcomings. System development model mature MVC and Struts 2 under Hibernate, Spring and Ajax technology, traditional B / S structure of the "dumb" client into a rich client, build asynchronous communication, refresh-free web applications, HTTP request to reduce the wait for the phenomenon appears to improve efficiency in the implementation of Web applications. Extjs use powerful UI components to improve the user experience and to build a more rational and more efficient Web application framework.

**【Key words】** object-relational mapping; dependency injection; oriented programming section; Ext JS; Ajax; AOP programming; ORM;

# 目录

|  |           |
|--|-----------|
| <b>第1章 绪 论</b> .....                           | <b>4</b>  |
| <b>1.1、 课题背景与现状</b> .....                      | <b>4</b>  |
| 1.1.1、 实验室管理平台的现状: .....                       | 4         |
| 1.1.2、 我校护理实验室建设现状: .....                      | 4         |
| 1.1.3、 课题来源: .....                             | 4         |
| <b>1.2、 开发环境及运行环境</b> .....                    | <b>5</b>  |
| <b>第2章 技术简介</b> .....                          | <b>6</b>  |
| <b>2.1、 AJAX 技术简介</b> .....                    | <b>6</b>  |
| 2.1.1、 概述.....                                 | 6         |
| 2.1.2、 Ajax 的工作原理 .....                        | 6         |
| <b>2.2、 SSH 框架简介</b> .....                     | <b>7</b>  |
| 2.2.1、 Struts2 框架 .....                        | 7         |
| 2.2.2、 Hibernate 框架 .....                      | 9         |
| 2.2.3、 Spring 框架 .....                         | 10        |
| <b>2.3、 开源框架与 EJB 对比</b> .....                 | <b>11</b> |
| 2.3.1、 框架对比.....                               | 11        |
| 2.3.2、 结论 (Conclusions) .....                  | 12        |
| <b>第3章 系统设计</b> .....                          | <b>13</b> |
| <b>3.1、 系统的功能介绍</b> .....                      | <b>13</b> |
| 3.1.1、 系统功能模块简介 .....                          | 13        |
| 3.1.2、 系统整体架构图.....                            | 13        |
| <b>3.2、 AJAX 界面的用户体验设计</b> .....               | <b>13</b> |
| <b>3.3、 软件编码的层次设计</b> .....                    | <b>14</b> |
| <b>第4章 系统实现</b> .....                          | <b>16</b> |
| <b>4.1、 EXT 框架的应用</b> .....                    | <b>16</b> |
| 4.1.1、 Extjs 之主界面布局.....                       | 16        |
| 4.1.2、 Ext Grid 组件 .....                       | 17        |
| <b>4.2、 PPT 在线网页处理</b> .....                   | <b>18</b> |
| <b>4.3、 CSS 界面控制实现</b> .....                   | <b>20</b> |
| 4.3.1、 Freemake 分页实现.....                      | 20        |
| <b>第5章 设计模式的应用</b> .....                       | <b>23</b> |
| <b>5.1、 基于工厂模式实现的 DAO 实现</b> .....             | <b>23</b> |
| <b>5.2、 DAO 操作封装的 template method 模式</b> ..... | <b>23</b> |
| <b>第6章 总结与展望</b> .....                         | <b>25</b> |
| <b>6.1、 总结</b> .....                           | <b>25</b> |

|                 |    |
|-----------------|----|
| 6.2、 前景展望 ..... | 25 |
| 参考文献 .....      | 26 |

## 第1章 绪 论

### 1.1、课题背景与现状

#### 1.1.1、实验室管理平台的现状：

根据国家信息化建设规划的要求，随着互联网技术的不断发展和普及，各种办公自动化管理软件已出现在各级政府部门以及各类企事业单位的管理平台上。在国际上，各类实验室均已开始朝网络化管理的方向发展并日臻成熟。把信息技术与实验室现有的质量保证体系相结合，旨在优化实验室的业务流程，提高试验信息传递的时效性、数据完整化和利用综合化，降低试验人员工作量，降低实验室管理成本，通过精简实验室现有的运作环节，探索信息技术与实验室质量管理体系有机结合的新型管理模式，以提升实验室的综合竞争力。

目前的实验室检测系统，处于深化市场机制的过程中，还未采用各种现代化管理手段，作为实验室主管，无法快速、全面、准确地掌控试验收入、合同状况、试验进度、人员管理等实验室信息；人员和任务分配过程较复杂；检验任务书、试验报告、原始记录等信息需要重复录入，而且查询、生成不方便；实验仪器设备的查询、维修、校准、各种标准文本的发放、查询等管理手续繁琐；从检验任务书的传递、检验，以及检验报告等都由人工处理；虽然各部门都配备了电脑，但是大多数部门的计算机都是独立使用，没有很好地实现资源共享。

#### 1.1.2、我校护理实验室建设现状：

我校护理实验室之前虽然已建设网站，但其信息容量太小，只有一些可以浏览、下载功能的页面，如规章制度、中心介绍、最新动态、通知等，网站的作用只是起到对部门的宣传，由于没有交互性，用户有可能对有些说明不理解或者理解有误等原因而对工作中的某些细节执行有误，最终为了准确性大部分工作中还是会通过传统的以手工为主的方式来处理问题。

#### 1.1.3、课题来源：

本课题《基于 AJAX 技术的护理实验室平台设计与实现》是针对当前我校护理实验室网站存在的问题：信息容量小，交互性差等问题而专门设计的。

本系统采用将成熟的开发模式 MVC 下的 Struts 2 和 Hibernate、Spring 与 Extjs Ajax 技术相结合，让其协同工作，提高 B / S 模式 Web 应用程序的响应速度，改善用户体验，以构建更高效更合理的 Web 应用框架，提高实验平台的交互性。

### 1.2、开发环境及运行环境

本系统用面向对象方法进行系统分析和设计。系统采用三层架构，实现表示层，业务逻辑层和数据持久层的分离。选用 JEE 开源框架(struts,spring,hibernate)结合 AJAX 技术开发，数据库建模使用 PowerDesigner 建模工具，以 MS SQL Server2000 为后台数据库，以 Eclipse 加 MyEclipse 插件作为开发环境(IDE)，以 JAVA 作为主要开发语言来实现。WEB 服务器采用 Tomcat6.0.18。

## 第2章 技术简介

### 2.1、AJAX 技术简介

#### 2.1.1、概述

从本质上来说, Ajax 并不是什么新技术, 它综合应用了 JavaScript、XHTML 与 CSS、DOM、XML 与 XSTL、XMLHttpRequest 等技术, 为用户提供了无刷新的动态数据交换等功能。Ajax 的核心是 JavaScript 对象 XMLHttpRequest。XMLHttpRequest 处理所有服务器通信的对象, 是一种支持异步请求的技术。简而言之, XMLHttpRequest 使应用程序可以使用 JavaScript 向服务器提出请求并处理响应, 而不阻塞用户。在这种模式下, 服务器将原来直接发送到用户表现层的数据, 改为向 Ajax 引擎返回其可用数据, 这些数据可以是纯文本、XML 或者需要的其他格式, 唯一的要求就是 Ajax 引擎能够理解和解析这些数据, 当 Ajax 引擎收到这些服务器的响应后, 将会触发一些操作, 即完成数据的解析工作以及由于其所提供的数据对用户的界面进行必要的修改。由于在这个过程中传送的信息比传统模式下要少许多, 因此用户界面的更新速度明显加快, 改善用户的体验。

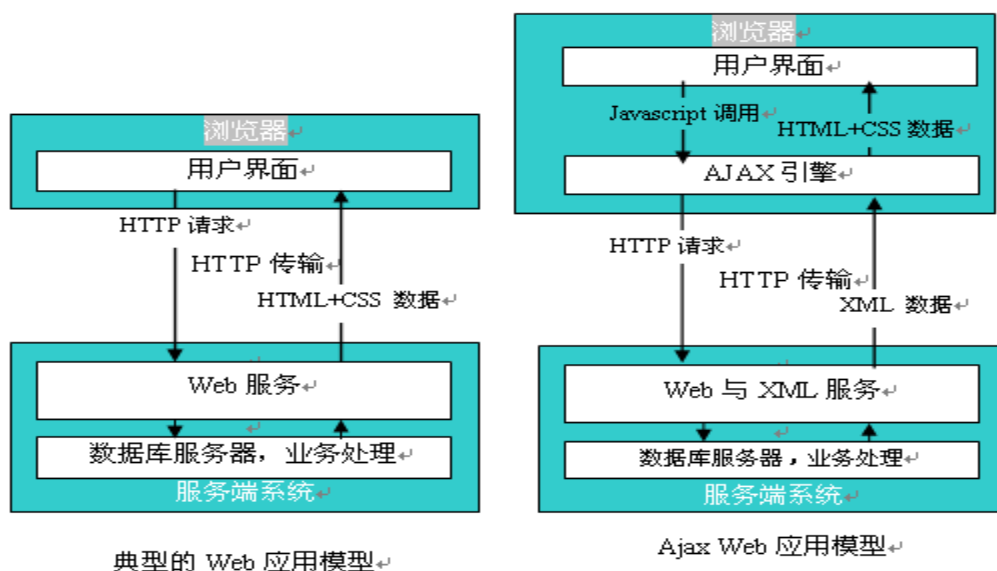


图 1 传统 Web 应用模型(左)与 Ajax 模型的比较(右)

#### 2.1.2、Ajax 的工作原理

传统的 web 应用遵循一种请求 / 响应的模式, 对于每个请求都会重新加载整个页面。Ajax 的工作原理相当于在用户和服务器之间加了一个中间层, 使用户操作与服务器响应异步化, 从而消除了网络交互过程中的“处理—等待—处理”的缺点。每当需要更新时, 客户端 Web 页面的修改是异步的和逐步增加的。并且客户端的浏览器可以分担一部分的业务逻辑, 而不再仅仅是显示内容; 服务器传送的仅仅是数据, 而不是内容。这样, Ajax 在提交 Web 页面内容时大大提高了用户界面响应的速度。在基于 Ajax 的应用程序中没有必要长时间等待整个页面的刷新。页面中需要更新的那部分才进行更改, 如果可能的话, 更新是在本地完成的, 并且是异步的。所以基于 Ajax 技术的应用程序可以给用户提供桌面程序般的使用体验。由于 Ajax 技术本身的优越性, 基于 Ajax 技术的 Web 型框架现在已经应用到很多程序中。例如 Gmail、Google Map、Google Suggest 就是几个经典的 Ajax 应用。

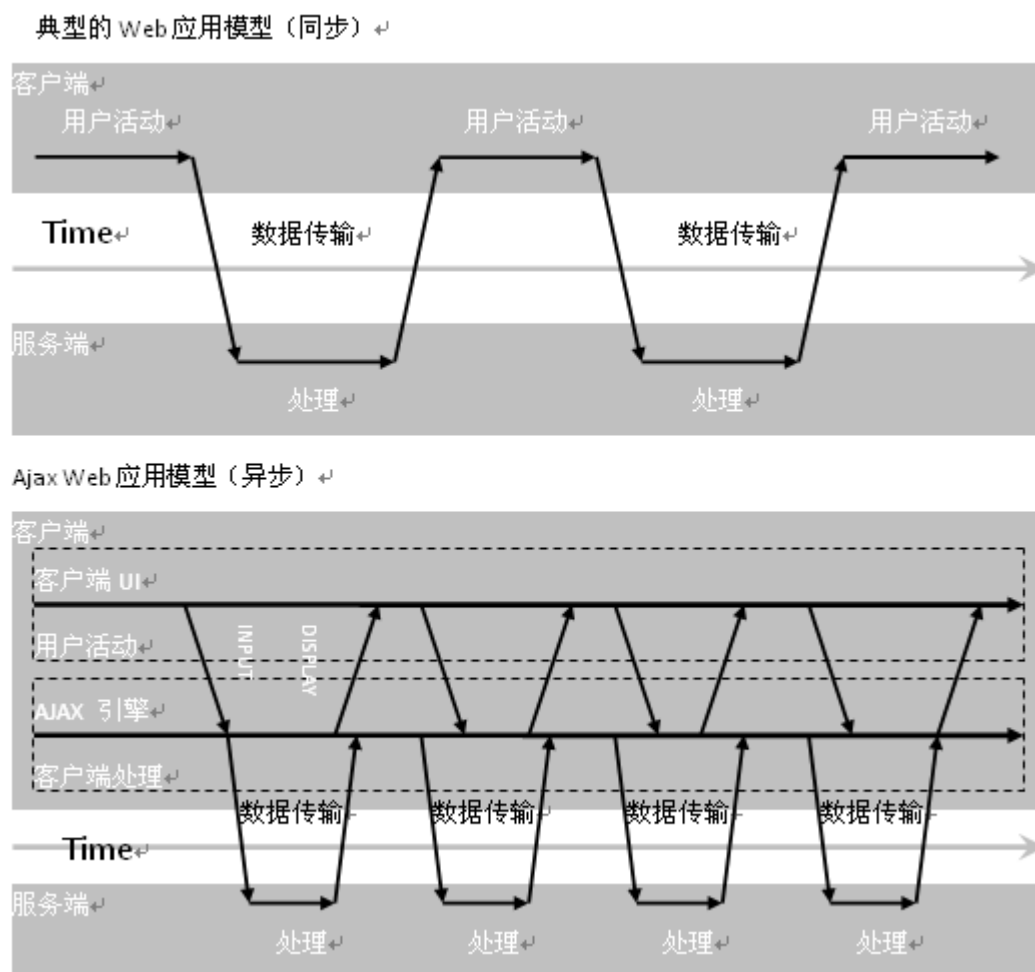


图 2 传统 Web 应用的同步交互过程(上)和 Ajax 应用的异步交互过程的比较(下)

## 2. 2、SSH 框架简介

### 2. 2. 1、Struts2 框架

#### 1) MVC 概述

MVC (Model-View-Controller, 模型—视图—控制器模式) 用于表示一种软件架构模式。它把软件系统分为三个基本部分: 模型(Model), 视图(View)和控制器(Controller)。

**模型:** 模型代表了应用程序的核心功能, 它封装了应用程序的数据结构, 集中体现了应用程序的状态。它负责处理用户的数据, 实现业务逻辑。同时, 模型还为视图的显示提供数据, 并被多个视图所共用。但它并不了解视图和控制器的信息。

**视图:** 视图主要是指与用户的界面, 也即应用程序的外观。视图可以接受用户的输入, 并将数据转交给控制器。同时, 视图还负责展现模型传递给用户的数据, 当后台模型更新数据时, 视图也应随之更新它的显示。

**控制器:** 控制器负责接收用户的请求和数据, 接着判断将请求和数据交由哪一个模型来处理, 最后调用视图来显示模型返回的数据。它将模型和视图联系在一起, 是整个 WEB 应用运行的调度者。



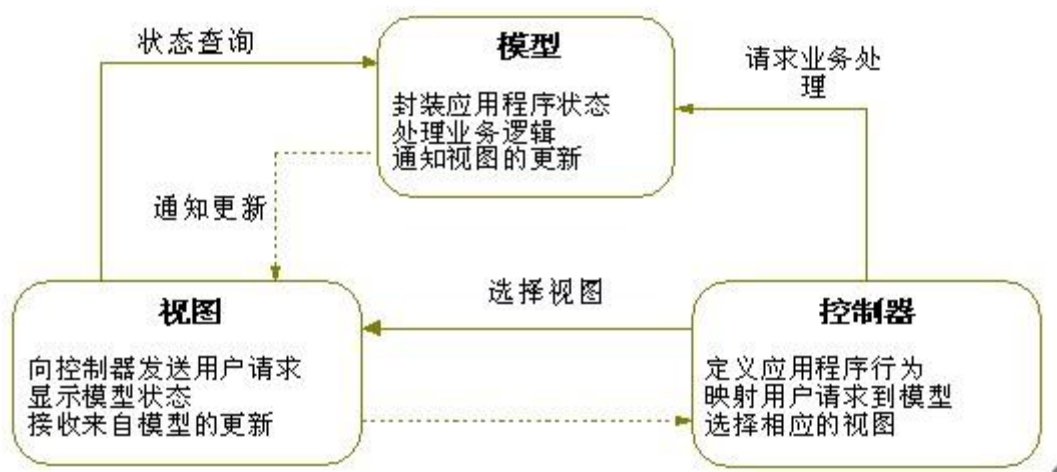


图 3 MVC 架构

## 2) Struts 2 概述

Struts 2 使用了 WebWork 的设计核心，它同样使用拦截器作为处理（Advice），以用户的业务逻辑控制器为目标，创建一个控制器代理。控制器代理负责处理用户请求，处理用户请求时回调业务控制器的 execute 方法，该方法的返回值决定 Struts2 呈现给用户怎样的视图资源。

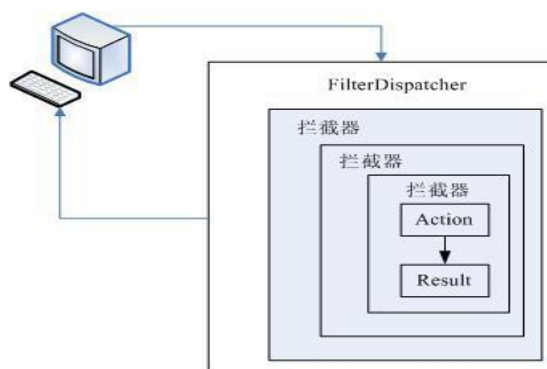
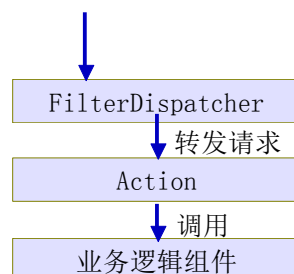


图 4 Struts2 的体系概图

Struts 框架由 3 个部分组成：核心控制器 FilterDispatcher，业务控制器和用户实现的业务逻辑组件。

### 核心控制器 FilterDispatcher

FilterDispatcher 是 Struts2 框架的核心控制器，该控制器作为一个 Filter 运行在 Web 应用中，它负责拦截所有的用户请求，当用户请求到达时，该 Filter 会过滤用户请求。如果用户请求以 action 结尾，该请求将被转入 Struts2 框架处理。



### 业务控制器

业务控制器组件就是用户实现的 Action 类的实例，Action 类里通常包含了一个 execute 方法，该方法返回一个字符串——该字符串就是一个逻辑视图名。配置 Struts2 的 Action 需要如下

三部分的定义：Action 所处理的 URL；Action 组件所对应的实现类；Action 里包含的逻辑视图和物理资源之间的对应关系。

### 视图组件

Struts2 除了能使用 JSP 作为视图技术外,还可以使用其它的模板技术,如 FreeMaker, Velocity 作为视图技术。总而言之, Struts 框架的大致处理流程如下:

- 1.浏览器发送请求,例如请求/reports/myreport.pdf、/mypage.action 等。
- 2.核心控制器 FilterDispatcher 根据请求决定调用合适的 Action。
- 3.Struts 的拦截器链自动对请求应用通用功能,例如 validation 或文件上传功能。
- 4.回调 Action 的 execute 方法。
- 5.Action 的 execute 方法处理结果信息被输出到浏览中,可以是 jsp 页面、图像,也可以是 pdf 文档等其它文档。

### 2.2.2、Hibernate 框架

对象-关系映射 (Object/Relation Mapping, 简称 ORM), 是随着面向对象的软件开发方法发展而产生的。对象-关系映射(ORM)系统一般以中间件的形式存在, 主要实现程序对象到关系数据库数据的映射。

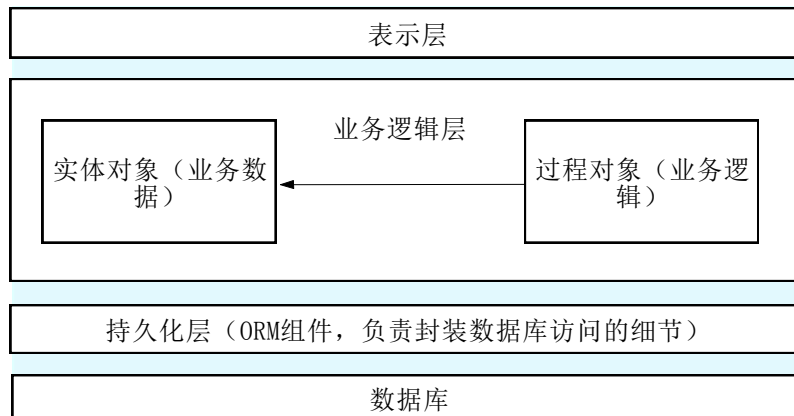


图 5 ORM 模式

Hibernate 是一个开放源代码的对象关系映射框架,它对 JDBC 进行了非常轻量级的对象封装,使得 Java 程序员可以随心所欲的使用对象编程思维来操纵数据库。Hibernate 可以应用在任何使用 JDBC 的场合,既可以在 Java 的客户端程序使用,也可以在 Servlet/JSP 的 Web 应用中使用,最具革命意义的是,Hibernate 可以在应用 EJB 的 J2EE 架构中取代 CMP,完成数据持久化的重任。

Hibernate 的核心接口一共有 5 个,分别为:Session、SessionFactory、Transaction、Query 和 Configuration。

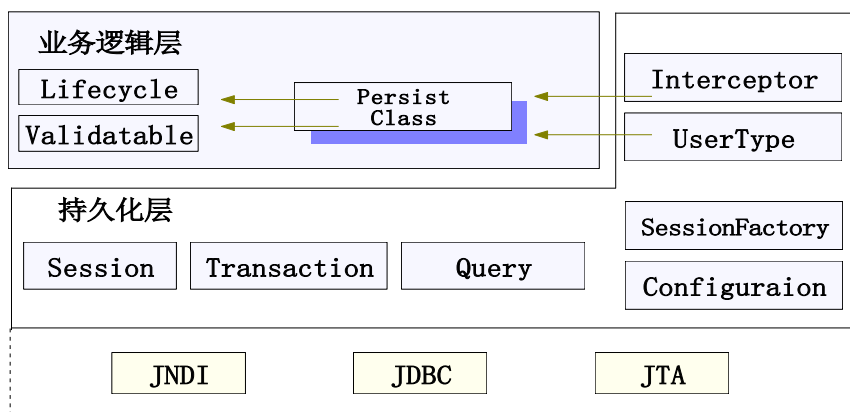


图 6 Hibernate 核心接口

**Session 接口:**Session 接口负责执行被持久化对象的 CRUD 操作(CRUD 的任务是完成与数据库的交流, 包含了很多常见的 SQL 语句。)。但需要注意的是 Session 对象是非线程安全的。同时, Hibernate 的 session 不同于 JSP 应用中的 HttpSession。这里当使用 session 这个术语时, 其实指的是 Hibernate 中的 session, 而以后会将 HttpSession 对象称为用户 session。

**SessionFactory 接口:**SessionFactory 接口负责初始化 Hibernate。它充当数据存储源的代理, 并负责创建 Session 对象。

**Configuration 接口:**Configuration 接口负责配置并启动 Hibernate, 创建 SessionFactory 对象。

**Transaction 接口:**Transaction 接口负责事务相关的操作。

**Query 和 Criteria 接口:**Query 和 Criteria 接口负责执行各种数据库查询。它可以使用 HQL 语言或 SQL 语句两种表达方式。

### 2.2.3、Spring 框架

#### 1) Spring 概述

Spring 是一个轻量级的开源框架。它包括多方面的特性, 并很好地被组织在下图所示的七个模块中。

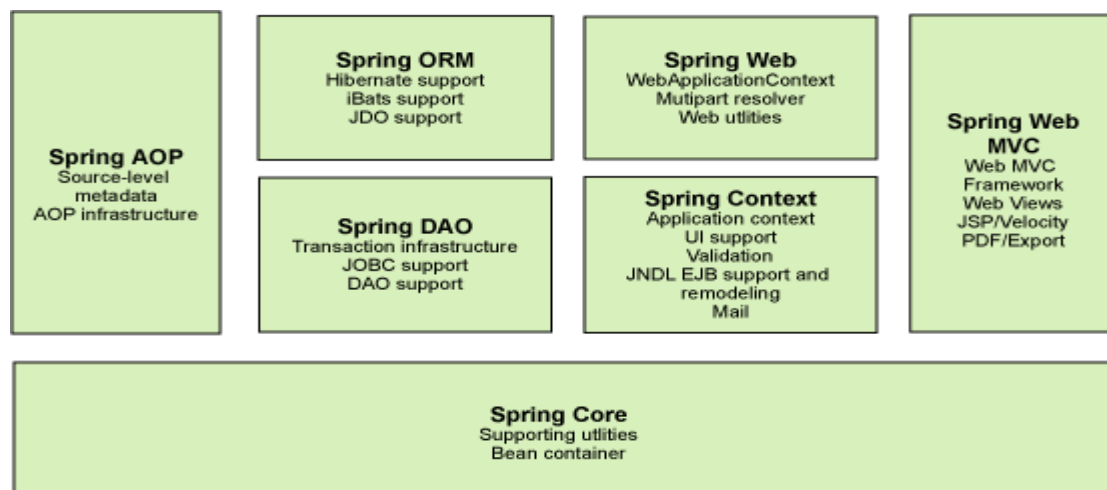


图 7 Spring 框架概述

#### 2) 控制反转容器

##### 容器和 bean 的基本原理

简单地讲, bean 就是被 Spring 容器初始化、装配及被管理的对象。在 Spring 中, BeanFactory 是 IoC 容器的核心接口。Spring 提供了许多易用的 BeanFactory 实现, XmlBeanFactory 是最

常用的一个。该实现以 XML 方式描述组成应用的对象以及对象间的依赖关系，从而构建一个完全可配置的系统或应用。

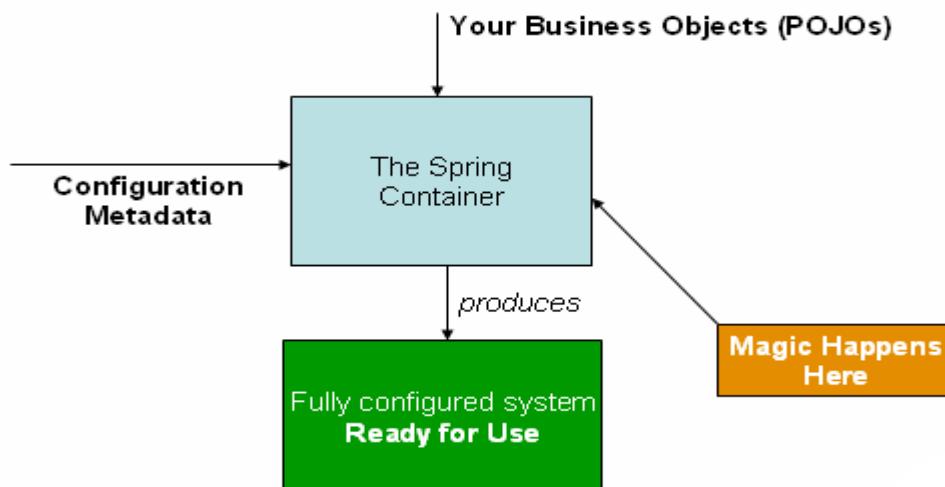


图 8 Spring ioc 容器

### 配置元数据

Spring Ioc 容器将读取配置元数据，并通过它对应用中的各个对象进行实例化，配置以及组装。通常情况下我们使用简单直观的 XML 来作为配置元数据的描述格式。

### 注入依赖

依赖注入（DI）背后的基本原理是对象之间的依赖关系。DI 主要有两种注入方式，即 Setter 注入和构造器注入。

## 2. 3、开源框架与 EJB 对比

### 2. 3. 1、框架对比

Spring 是一个流行的但非标准的开放源码的框架。它主要由 Interface21 Inc. 开发和支配。它是基于 DI（Dependency Injection：依赖注入）设计模式而架构的。Spring 可以单独工作，也可以与现存的应用服务器协同工作，它很好的利用了 XML 配置文件。EJB 3.0 是一个由 JCP（Java Community Process）定义的标准框架，并被所有主要 J2EE 厂商支持。EJB 3.0 规范的最先版本的源码和商业实现已经可以从 JBoss 和 Oracle 得到。EJB 3.0 规范大量使用了 Java 注释技术。

这两个框架它们拥有一个相同的核心设计理念：把中间件服务传递给松散耦合的简单旧式 Java 对象（POJO）。框架通过截取执行上下文环境将应用服务绑定到 POJOs，或者在运行时间将服务对象注入到 POJOs。POJOs 自身并不关心如何绑定并很少依赖于框架。最终，开发者就可以专心于业务逻辑的设计和单元测试它们的 POJOs（单元测试并不需要此框架）。另外，由于 POJOs 并不一定要继承框架类或者实现框架接口，所以开发者拥有高度的弹性去创建他们自己的继承结构和构造应用程序。下图为两框架的比较

| 框架         | EJB2/EJB3  | Spring Framework 1.x            |
|------------|--|---------------------------------|
| 灵活性(松耦合)   | EJB3 比 EJB2 更具灵活性, EJB3 支持应用系统 POJO              | 支持应用系统 POJO, 框架本身可分离配置          |
| 功能完整性      | 全面, 支持异步 JMS 分布式事务                               | 较为全面。有自己的表现层和持久层模板, 可支持异步       |
| 领域范围       | 支持业务逻辑 Session                                   | 不支持, 需要开发者额外基于 ThreadLocal 编制代码 |
| IoC/AOP 支持 | EJB3 支持 IoC, JBoss 等 EJB3 服务器支持 AOP; 基于业务组件的较粗粒度 | 基于 JavaBeans 类的细粒度支持 AOP        |
| 单台性能       | 一般, 批量查询等大数据量业务处理须小心, 存在本地不透明缺陷。                 | 一般, 应用程序可配置 cache/Pool 以提高性能    |
| 可伸缩性       | 可支持多台服务器分布式计算。                                   | 不支持, 可依靠 EJB 实现                 |
| 开发效率       | 学习曲线长, 导致熟练掌握难。借助商业开发工具可加快熟练者的开发速度。              | 可挑选只适合自己的功能实现。相对 EJB 稍简单。       |
| 系统规模       | EJB2 适合大型系统, EJB3 适合中大型系统                        | 适合中小型系统, 可借助 EJB 支持中大型系统        |

表 1 EJB VS Spring

## 2.3.2、结论 (Conclusions)

尽管 Spring 和 EJB 3.0 都旨在于为松耦合的 POJOs 提供企业服务, 但是它们却使用了截然不同的方法去实现这个目标。它们都非常重视并且大量使用了依赖注入设计模式。EJB 3.0 是基于标准的方法, 广泛的使用注释技术和应用程序服务的紧密集成导致更多的厂商独立性和开发效率。Spring 配合使用依赖注入和集中式的 XML 配置文件使得开发者能够构建出更富有弹性的应用程序, 并且能够一次性与几个由不同提供者所提供的应用服务同时工作。

## 第3章 系统设计

### 3.1、系统的功能介绍

#### 3.1.1、系统功能模块简介



图 9 系统整体模块

#### 3.1.2、系统整体架构图

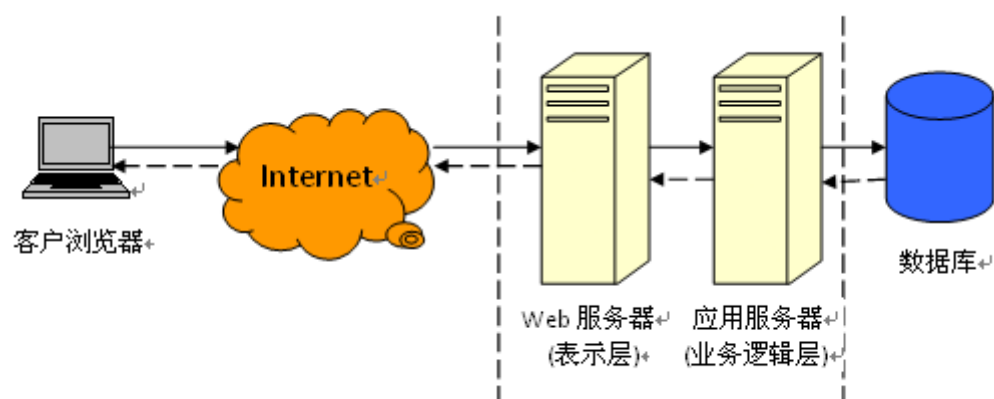


图 10 系统整体架构图

本系统为经典的 B / S 系统，系统采用三层架构，实现表示层，业务逻辑层和数据持久层的分离。这样分层可以方便后期维护。

### 3.2、AJAX 界面的用户体验设计

本系统采用 ExtJs 为后台的前端，利用 ExtJS 强大的 UI 组件与 Ajax 功能，因为后台不面向用户，只面向内部人员，访问量比较小，加载速度要求不高。之所以后台要用 EXT，是因

为后台对数据的操作比较多,经常要用到 DataGrid,正好 EXT 比较成熟的框架,在数据操作方面的功能非常强大,正满足需要。Ajax 避免了传统 web 解决方案中的提交/等待方式,实现了局部刷新,加快了信息显示的速度,将能更好地发挥各自的优势,能恰到好处地运用于 WEB 开发中,给开发者和用户分别以不同的技术角度和焕然一新的良好体验,有利于充分地体现 WEB 2.0 的特征与优势。

### 3.3、软件编码的层次设计

为了充分体现 MVC 架构,系统编码实现表示层,业务逻辑层和数据持久层的分离。这样分层的优势主要是:低耦合性。视图层和业务层分离,这样就允许更改视图层代码而不用重新编译模型和控制器代码,同样,一个应用的业务流程或者业务规则的改变只需要改动 MVC 的模型层即可。因为模型与控制器和视图相分离,所以很容易改变应用程序的数据层和业务规则。高重用性和可适用性。随着技术的不断进步,现在需要用越来越多的方式来访问应用程序。MVC 模式允许你使用各种不同样式的视图来访问同一个服务器端的代码。较低的生命周期成本。MVC 使降低开发和维护用户接口的技术含量成为可能。快速的部署。使用 MVC 模式使开发时间得到相当大的缩减,它使程序员(Java 开发人员)集中精力于业务逻辑,界面程序员(HTML 和 JSP 开发人员)集中业务于表现形式上。可维护性。分熟视图层和业务逻辑层也使得 WEB 应用更易于维护和修改。

以下是各层的介绍

#### 1) 表示层框架 Struts 2

Struts2 是一个在 JSP Model2 基础上实现的 MVC 框架,主要分为模型(Model)、视图(Viewer)和控制器(Controller)三部分,其主要的设计理念是通过控制器将表现逻辑和业务逻辑解耦,以提高系统的可维护性、可扩展性和可重用性。如下是 Struts 2 在 web.xml 中的具体配置

```
<!-- end configuration for struts 2.x -->
<filter>
    <filter-name>struts2</filter-name>
    <filter-class>org.apache.struts2.dispatcher.FilterDispatcher
</filter-class>
</filter>
<filter-mapping>
    <filter-name>struts2</filter-name>
    <url-pattern>*.action</url-pattern>
</filter-mapping>
<!-- start configuration for struts 2.x -->
```

#### 2) 业务逻辑层框架 Spring

Spring 是一个解决了许多 J2EE 开发中常见问题并能够替代 EJB 技术的强大的轻量级框架。spring 作为业务逻辑,用来管理 bean.它负责把表示层 action, 业务 javabean,dao 层的 bean 粘合起来。如下是 Spring 管理 Bean 的配置

```
<!--定义了 Hibernate 的 SessionFactory -->
<bean
    id="sessionFactory"
    class="org.springframework.orm.hibernate3.LocalSessionFactoryBean">
    <property name="lobHandler" ref="lobHandler"/>
    <property name="configLocation">
        <value>classpath:config/hibernate/hibernate.cfg.xml</value>
```



```
</property>
</bean>
```

### 3) 数据持久层框架 Hibernate

O/R mapping 技术是为了解决关系型数据库和面向对象的程序设计之间不匹配的矛盾而产生的。Hibernate 是目前最为流行的 O/R mapping 框架，它在关系型数据库和 Java 对象之间做了一个自动映射，使得程序员可以以非常简单的方式实现对数据库的操作。具体配置如下

```
<hibernate-configuration>
  <session-factory name="sessionFactory">
    <property name="hibernate.connection.driver_class">
com.microsoft.jdbc.sqlserver.SQLServerDriver</property>
    <property name="hibernate.connection.url">
jdbc:microsoft:sqlserver://localhost:1433;DatabaseName=test;SelectMethod=cursor</property>
    <property name="hibernate.connection.username">sa</property>
<property name="hibernate.connection.password">123456</property>
<property
name="hibernate.dialect">org.hibernate.dialect.SQLServerDialect
</property>
  </session-factory>
</hibernate-configuration>
```



## 第4章 系统实现

### 4.1、EXT 框架的应用

#### 4.1.1、Ext.js 之主界面布局

在传统的 B/S 系统实现中，我们一般都会采用 frame 来把主页面分成几个部分，如下面代码就是采用了 frame 的实现的主页面：

```
<frameset rows="100,*,24 " framespacing="0">
  <frame noresize="noresize" target="contents" name="head"
    src="pages/head.jsp" scrolling="no">
  <frameset cols="180,*">
    <frame name="left" src="pages/left.jsp" marginheight="0"
      marginwidth="0" scrolling="no" noresize />
    <frame name="right" src="pages/main.jsp" marginheight="0"
      marginwidth="0" scrolling="yes" noresize />
  </frameset>
  <frame name="foot" src="pages/foot.jsp" marginheight="0"
    marginwidth="0" scrolling="no" noresize frameborder="0" />
</frameset>
```

这个就是采用 frame 和 frameSet 把网页放在 head、left、main 和 foot 四个部分。用 ext 的话我们则不需要用 frame，而改用 Ext 自带的布局对象 Panel 与 Viewport 来进行布局，并且布局 -border: 将容器分为五个区域: east, south, west, north, center，本系统布局代码如下

```
//init main page tables
var panel = new Ext.Panel({
  id:'main-panel',
  baseCls:'x-plain',
  renderTo: Ext.get("center"),
  layout:'table',
  layoutConfig: {columns:2},
  defaults: {frame:true, width:395, height: 300},
  items:[{
    title:'公告信息',
    colspan:2,
    collapsible:true,
    contentEl:'afficheDiv'
  }]
});

var viewport = new Ext.Viewport({
  layout:'border',
  items:[
    {
      region: 'north',
```

```

        border: false,
        height:0,
    },{
        region:'west',
        id:'west-panel',

        layout:'accordion', //类似 outlook bar 的效果

        layoutConfig:{
            animate:true    //内部布局变动会显示动画
        },
        items: [{
            title:'首页管理',

            html: Ext.getDom('newMenus').innerHTML,
            border:false,
            autoScroll:true,
            iconCls:'unit'
        }, {...}]
    },tabPanel]
});


```

下图就是本系统用 Ext UI 经布局后的后台主界面



#### 4.1.2、Ext Grid 组件

Ext 的 Grid 控件能够支持多种数据类型，如二维数组、Json 数据和 XML 数据，甚至包括我们自定义的数据类型。Ext 为我们提供了一个桥梁 Ext.data.Store，通过它我们可以把任何格式的数据转化成 grid 可以使用的形式，这样就不需要为每种数据格式写一个 grid 的实现了。

| 添加  删除  |                          |           |                     |      |      |     |
|---|--------------------------|-----------|---------------------|------|------|-----|
|   | <input type="checkbox"/> | 标题 ▲      | 发布时间                | 管理员  | 类别 ▼ | 浏览量 |
| 1   | <input type="checkbox"/> | 七国集团财长宣布对 | 2008-10-11 14:00:16 | leaf | 动态新闻 | 4   |
| 2   | <input type="checkbox"/> | 动态新闻      | 2008-10-11 14:14:08 | leaf | 动态新闻 | 0   |
| 3   | <input type="checkbox"/> | 动态新闻      | 2008-10-11 14:14:13 | leaf | 动态新闻 | 0   |
| 4   | <input type="checkbox"/> | 动态新闻      | 2008-10-11 14:14:20 | leaf | 动态新闻 | 0   |
| 5   | <input type="checkbox"/> | 动态新闻      | 2008-10-11 14:14:38 | leaf | 动态新闻 | 16  |
| 6   | <input type="checkbox"/> | 动态新闻      | 2008-10-11 14:15:07 | leaf | 动态新闻 | 0   |
| 7   | <input type="checkbox"/> | 动态新闻      | 2008-10-11 14:15:28 | leaf | 动态新闻 | 0   |
| 8   | <input type="checkbox"/> | 动态新闻      | 2008-10-11 14:15:48 | leaf | 动态新闻 | 4   |
| 9   | <input type="checkbox"/> | 动态新闻动态新闻  | 2008-10-11 14:14:31 | leaf | 动态新闻 | 2   |
| 10  | <input type="checkbox"/> | 税务总局人士称正在 | 2008-10-11 14:13:29 | leaf | 动态新闻 | 1   |

```

var sm = new Ext.grid.CheckboxSelectionModel({handleMouseDown:Ext.emptyFn});
var cm = new Ext.grid.ColumnModel([
    new Ext.grid.RowNumberer(),

    sm,{header:'标题',dataIndex:'newsTitle',editable:false},

    {header:'发布时间',dataIndex:'newsPbtime',editable:false},

    {header:'管理员',dataIndex:'admName',editable:false},

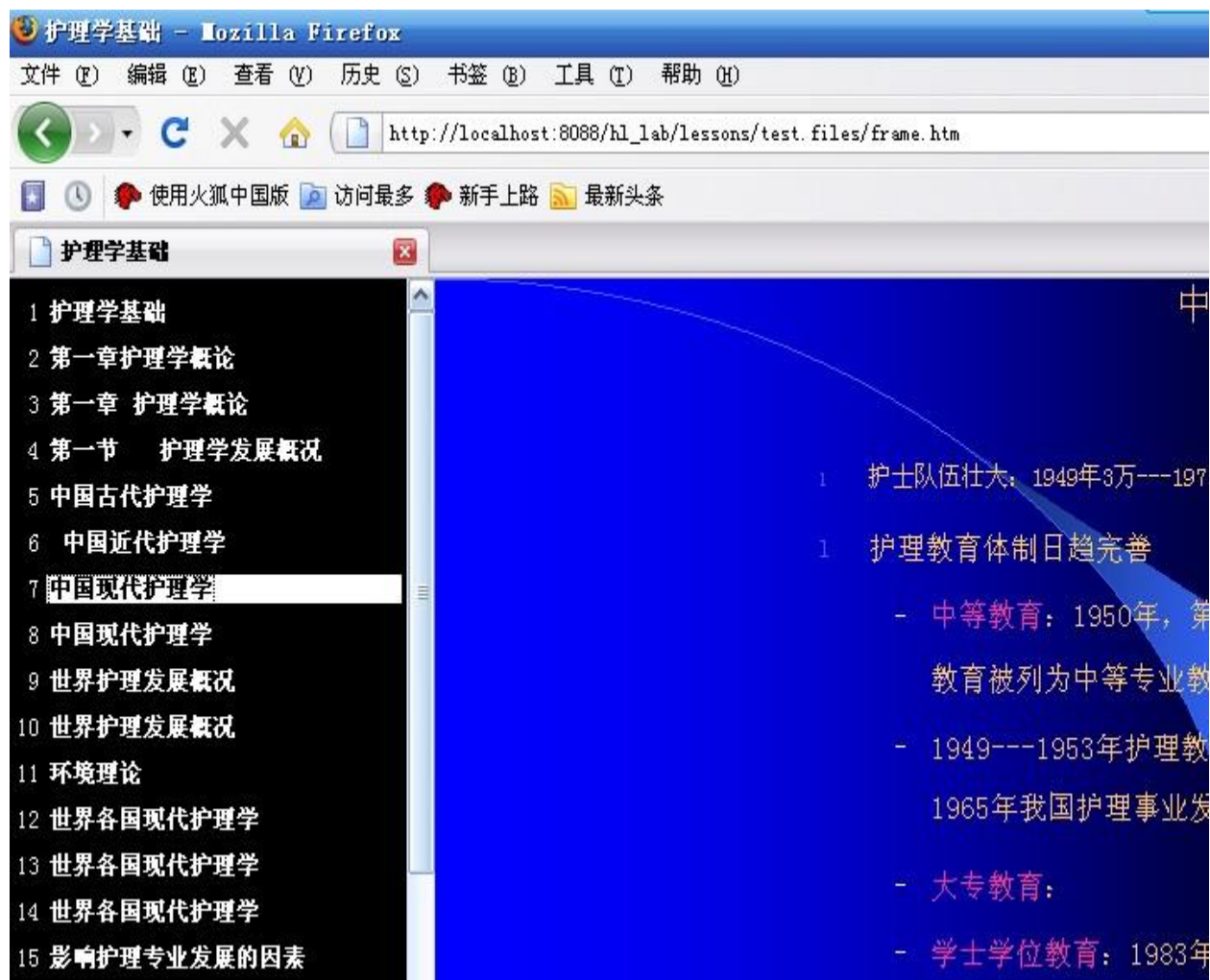
    {header:'类别',dataIndex:'wtName',editable:false},

    {header:'浏览量',dataIndex:'newsNodnum',editable:false},

    {header:'操作',dataIndex:'newsId',editable:false, renderer:renderShowNews}
]);
var newListDataStore = new Ext.data.Store({
    proxy : new Ext.data.HttpProxy({
        url
        :
        "../adminPath/adminnewaction.action?newsTypeInfo="+node.href.split("/").reverse()[0]+"",
        method : "POST"
    }),
    reader : new Ext.data.JsonReader({
        totalProperty : "totalCount",
        root : "newslst"
    }, newslstDataType)
});

```

#### 4. 2、PPT 在线网页处理



PPT 在线网页是指将 PPT 文件保存为 Web 格式 ,用户在 Web 浏览器即可方便的阅读 PPT 内容。系统管理员可以将 PPT 文件另存为网页格式(htm ,html) ,这样会产生一个 htm 文件跟一个与之对应的.files 文件夹 ,将它们压缩成压缩包。用户将压缩包上传到服务器后 ,服务端将调用系统的 UNRAR 自动进行解压 ,关键代码如下 :

```
public static void unzip(String src, String dest) throws Exception {
    Runtime rt = Runtime.getRuntime();
    Process p = rt.exec("C:\\Program Files\\WinRAR\\UNRAR.EXE x -o+ -p- "+ src + " " + dest);
    StringBuffer sb = new StringBuffer();
    InputStream fis = p.getInputStream();
    int value = 0;
    while ((value = fis.read()) != -1) {
        sb.append((char) value);
    }
    fis.close();
}
```

```
String result = new String(sb.toString().getBytes("ISO-8859-1"), "GBK");
}
```

考虑到服务器存储容量有限，需对不用或过时的课件进行清理。如下是删除课件的代码：

```
public void deletePpt(String deleteIds, String deletePaths)throws DeleteException {
    String ids[]=deleteIds.split(",");
    String[] deletePath=deletePaths.split(",");
    StringBuilder sb=new StringBuilder();
    for(int i=0;i<ids.length;i++){
        sb.append(ids[i]);
        if(i < ids.length - 1 && ids.length > 1)
            sb.append(",");
    }
    if(sb.length()>0)
        coursewareDAO.makeTransientBypptIds(sb.toString());
    for(int i=0;i<deletePath.length;i++){
        String[] samePath=deletePath[i].split("\\.");
        FileUtil.delFile(deletePath[i]);
        FileUtil.delFolder(samePath[0]);
    }
}
```

#### 4. 3、CSS 界面控制实现

系统前台采用 CSS 控制界面的布局，这样可以让内容与表现分离,有了 CSS，网页的内容(XHMTL)与表现就可以分开了。表现的统一，可以使网页的表现非常统一，并且容易修改。将 CSS 文件独立出来，可以在日后的网站调整更方便。

```
#main{                                /*中间部分右边主要栏目*/

    width:555px;

    float:right;        /*根据父元素进行绝对定位*/

    right:0;
    top:0;
}
.editandpass{
    text-align:center;
    fong-size:12px;
}
#news ul{
    margin-top:10px;
    margin-left:5px;
    margin-bottom:10px;
    padding-left:15px;
}
```

##### 4. 3. 1、Freemaker 分页实现



中心概况

08-06-02

[首页](#)   [上一页](#)   [下一页](#)   [尾页](#)   页 1/1   共 1 条   20 条记录/页

FreeMarker 是一个模板引擎，一个基于模板生成文本输出的通用工具，使用纯 Java 编写，FreeMarker 被设计用来生成 HTML Web 页面，特别是基于 MVC 模式的应用程序。由于本系统有很多地方需要用到分页，设计一个可重用的分页模板可以避免代码的重复编写。如下就是本系统分页模板的实现

```

<#macro pageList acName>
    <#if (newsTypeInfo?exists)>
        <div id="fenye">
            <a href="{acName}.action?newsTypeInfo={newsTypeInfo}&&currage=1">
                首页</a>

            <#if (currage<=1)>上一页<#else>

                <a
href="{acName}.action?newsTypeInfo={newsTypeInfo}&&currage={currage-1}">
                    上一页</a>

            </#if>

            <#if (currage==totalpage)>下一页

            <#else>
                <a
href="{acName}.action?newsTypeInfo={newsTypeInfo}&&currage={currage+1}">
                    下一页 </a> </#if>

                <a
href="{acName}.action?newsTypeInfo={newsTypeInfo}&&currage={totalpage}">
                    尾页 </a>

                页${currage}/${totalpage}共${totalCount}条${pagesize}条记录/页

            </div>
        </#if>
    </#macro>

```



## 第5章 设计模式的应用

### 5.1、基于工厂模式实现的 DAO 实现

DAO 封装了数据库增删改查的业务逻辑，每一次对数据库的操作，DAO 对象都必须和数据库进行连接。由于本系统采用的是 Hibernate 组件，因此，实际上与数据库的连接和对数据库进行相关的操作都由 Hibernate 组件托管了。而 DAO 所要做的事情就是获取 Hibernate 连接数据库的 Session 对象。因此，每一个 DAO 实例在更新数据库的时候，就需要请求 Session 实例。并且，Hibernate 组件为了保证事务的完整性和一致性，强制系统在一个事务中只能产生一个 Session。

因此在这里系统设计了一个 HibernateUtil 类，这个类的设计实现采用了 Singleton（单件模式）和 Factory（工厂模式）模式。由于需要保证 Session 的唯一性，HibernateUtil 类定义了 SESSIONFACTORY 对象，由 SESSIONFACTORY 工厂对象构造 Session 对象。因而，SESSIONFACTORY 对象实现了单件模式和工厂模式。SESSIONFACTORY 对象在服务器启动时候，系统就直接实例化一个 SESSIONFACTORY 对象。以后任何 DAO 对象都直接使用该对象

//系统启动时运行的一个静态程序模块

```
static {  
    try {  
        // 可以使用不同的 Hibernate 配置文件  
        SESSIONFACTORY =  
            new Configuration().configure().buildSessionFactory();  
    } catch (HibernateException ex) {  
        log.error("创建 SessionFactory 失败，错误信息：", ex);  
        throw new RuntimeException(  
            "创建 SessionFactory 失败，错误信息：", ex);  
    }  
}
```

因为 SESSIONFACTORY 对象是一个公用实例，同时也是一个单件，因此，各个 DAO 对象都需要调用同一个 SESSIONFACTORY 对象，而生成的 Session 实例为事务操作提供了一次性连接服务。基于这个原因，程序就必须强制 Session 保证操作的事务性，必须采用 Java 线程控制。在这里，系统将 Session 封装在一个 ThreadLocal 对象中，由 ThreadLocal 对象保证了 Session 操作的事务性。同时，ThreadLocal 对象中还启动多个操作线程共同完成事务操作，并共享对象内的公用系统资源。

### 5.2、DAO 操作封装的 template method 模式

由于 DAO 是一个公用的数据处理模块，因此，系统将其提取出来，然后根据各种业务处理需要对其进行创建。具体类图如图 5-1 所示，DAO 作为一个公共类实现了对具体 PO 在持久层的增、删、改、查和装载（从数据库中的数据值到 PO 数据对象需要一个装载的操作）操作。但是对数据的封装（即创建 DAO）实际上是由调用 DAO 的两个子抽象类：DBActionBase（数据基类）和 PageActionBase（分页数据基类）完成的。这两个子类实际上是 2 个抽象模板类，依赖于 DAO 类，在创建 DAO 对象，调用增删改查函数的基础上，增加了数据验证，创建持久数据对象以及各自的处理逻辑函数。业务逻辑模块要使用这两个类的时候必须继承其中的一个，并实现其中抽象的函数。那么为什么要采用这种实现方式呢？



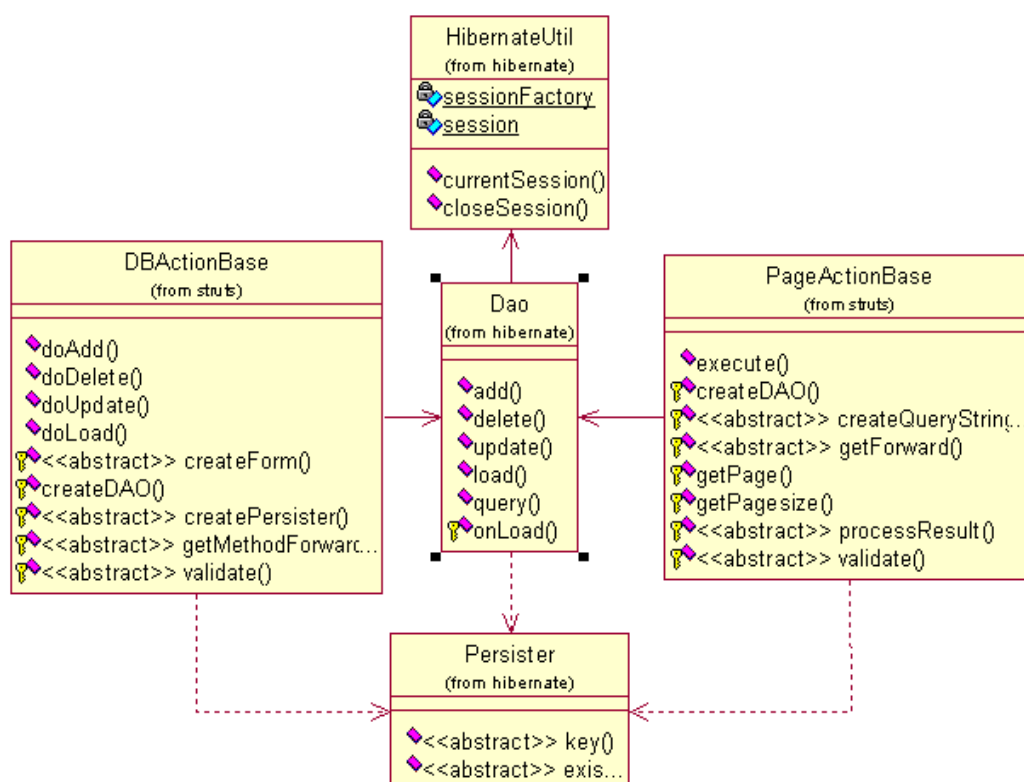


图 错误！文档中没有指定样式的文字。 -1 DAO 对象的上下文实现类图

这种实现策略就是经典的 **template 模式**（模板模式）。**template 模式**的基本思想就是：定义业务操作中的基本算法，实现公用方法，而将一些细节方法的实现延迟到子类中。**template method 模式**使得子类可以不改变一个算法的结构即可重定义算法的某些特定步骤。这个模式特别适合以下几种情况：

- （1）一次性实现一个算法的不变部分，并将可变的行为留给子类来实现。
- （2）各子类中公共的行为应被提取出来，并集中到一个公共父类中，以避免代码重复。识别现有代码中的不同之处，并且将不同之处分离为新的操作。最后，用一个调用这些新操作的模板方法来替换这些不同的代码。
- （3）控制子类的扩展。模板方法只在特定点调用“hook”操作，这样就只能允许在这些点进行扩展了。

因此，此处采用 **template method 模式**既可以先定义例如转向，创建 **DAO** 等函数先行进行定义，从而对继承类进行控制，又可以避免代码重复的问题。

## 第6章 总结与展望

### 6.1、总结

从本系统开始设计到现在，历时二个多月，到目前为止基本功能已经实现。本系统是在对传统

在这两个多月的项目开发过程中，我通过自学和查阅相关资料，对多个技术难题进行攻关并得到解决；掌握了许多新的知识，加深了对软件工程和团队开发的认识，特别是在软件部署方面积累了非常宝贵的经验。在开发的同时也得到了指导老师与同学们的关心、照顾和帮助。在此我对他们表示诚挚的谢意。

论文是在导师叶夏老师的指导下完成的。导师深厚的专业知识，严谨的治学态度，精益求精的工作作风，严以律己、宽以待人的崇高风范，平易近人的人格魅力对我影响深远。不仅使我树立了远大的学术目标、掌握了基本的研究方法，还使我明白了许多待人接物与为人处世的道理。本论文从选题到完成，每一步都是在导师的指导下完成的，倾注了导师大量的心血。在此，谨向导师表示崇高的敬意和衷心的感谢！

### 6.2、前景展望

通过本次毕业设计，我感到自己应用基础知识及专业知识解决问题的能力有了很大的提高，

并且这次毕业设计的选题，是一个实际的系统，因此，是在我即将工作之前，它是一次重要

演练。我想，通过这次毕业设计，到了工作单位后，我将能够更快的适应工作岗位和工作要

求。我对自己充满信心。总之，这次毕业设计对我而言是受益匪浅的。

## 参考文献

- [1]: MSDN 微软开发者网络资料库。http://msdn.microsoft.com/library
- [2]: ASP.NET 2.0 经典教程 CHRIS HART 等著 孟宪瑞、易磊译 人民邮电出版社 2006
- [3]: ASP.NET 2.0 高级编程 BILL EVJEN, SCOTT HANSELMAN 等著 李敏波译 清华大学出版社 2006
- [4]: ASP.NET 程序开发范例宝典 张跃廷、王小科、帖凌珍等著 人民邮电出版社 2006
- [5]: ASP.NET AJAX 程序设计 陈黎夫著 人民邮电出版社 2007
- [6]: AJAX IN ACTION DAVE CRANE, ERIC PASCARELLO 等著 人民邮电出版社 2006
- [7]: AJAX 模式与最佳实践 格罗斯(GROSS, C.) 著 李锟、张祖良、蔡毅、赵泽欣等译 电子工业出版社 2007
- [8]: FOUNDATIONS OF AJAX (美)RYAN ASLESON, NATHANIEL T. SCHUTTA 著 金灵等译 人民邮电出版社 2006
- [9]: PROFESSIONAL AJAX (美)NICHOLAS C. ZAKAS 著 徐锋、吴兰陟等译 人民邮电出版社 2006
- [10]: 征服 AJAX—WEB 2.0 开发技术详解 王沛, 冯曼菲著 人民邮电出版社 2006
- [11]: 征服 ASP.NET 2.0 AJAX—WEB 开发技术详解 陈冠军著 人民邮电出版社 2007
- [12]: JAVASCRIPT 高级程序设计 NICHOLAS C. ZAKAS 著 曹力、张欣等译 人民邮电出版社 2006
- [13]: JAVASCRIPT DOM 编程艺术 (英)JEREMY KEITH 著 杨涛, 王建桥, 杨晓云等译 人民邮电出版社 2007
- [14]: 征服 AJAX—WEB 2.0 快速入门与项目实践(.NET)施伟伟著 人民邮电出版社 2006
- [15]: 网站重构: 应用 WEB 标准进行设计 ZELDMAN, J 著 傅捷、王宗义、祝军等译 电子工业出版社 2004
- [16]: JAVASCRIPT 权威指南(第四版) DAVID FLANAGAN 著 张铭泽等译 机械工业出版社 2003
- [17]: 精通 JAVASCRIPT 动态网页编程 王俊杰著 人民邮电出版社 2007
- [18]: JAVASCRIPT 脚本程序设计 吴以欣、陈小宁著 人民邮电出版社 2005
- [19]: 网页设计基础—HTML, CSS 和 JAVASCRIPT 史晓燕、苏萍著 人民邮电出版社 2006
- [20]: HTML & XHTML 权威指南(第六版) (美)CBUCK MUSCIANO;BLL KENNEDY 著 张洪涛、邢璐等译 清华大学出版社 2007

## 附录

附录一 关键代码及界面截图

附录二 主要数据表