基于 Web 方式的学科馆员工作平台的设计与实现

王 峰 梁慧刚
（中国科学院国家科学图书馆武汉分馆　武汉　430071）

[**摘　要**] 本文介绍了中国科学院国家科学图书馆学科馆员工作平台项目建设的背景。阐述了如何使用 Struts 和 Hibernate 技术，设计并实现了一个基于Web的学科馆员工作平台系统。重点从表示层、业务层和持久层分析了实现的关键技术。

[**关键词**] 学科馆员 J2EE Struts Hibernate

The Design and Implement of Subject Librarian Platform Based on Web

Wang Feng, Liang Huigang
(The Wuhan Branch of National Science Library, CAS, Wuhan 430071, China)

【Abstract】 This paper firstly introduces the background of subject librarian platform. Then gives an effective solution about how to build a subject librarian platform by Struts and Hibernate. It focuses on implementation method on presentation tier, business tier and persistence tier.

【Key words】subject Librarian; J2EE; Struts; Hibernate

1 引言

进入 21 世纪以来，数字化环境对图书馆的发展形成了巨大的挑战，同时也提供了发展学科化个性化服务的良好的条件。当图书馆面对 google，百度等带来的冲击，面对着 e-science、e-learning、e-administration、e-media 带来的用户信息环境、用户信息行为、信息服务场景发生的根本性变化，图书馆需要认识到新的服务模式，新的组织机制，新的人员队伍才会适应用户需要，保障用户效果，促进用户能力持续增长，从而带动图书馆创新发展，使图书馆在新的信息环境中地位得到确立、稳固和提升。

基于对图书馆面临的新形势的充分理解，和对自身所具备优势和劣势的充分分析，中国科学院国家科学图书馆大力推行和发展了新型的学科化服务模式。中国科学院国家科学图书馆目前建立了专门的学科咨询服务部，探索建立具有创新性的学科馆员制度，充分发挥学科馆员的桥梁和纽带作用，面向科研一线为科研人员和课题研究小组提供个性化、学科化、知识化的创新服务。

中国科学院国家科学图书馆学科化服务推广以来，学科馆员的工作很大一部分已经需要在网络上开展，学科馆员也就迫切需要借助一个网络平台来开展信息交流、用户服务，进行虚拟团队管理。也正是基于此需求，国家科学图书馆将建设学科馆员交流平台作为国家科学图书馆 2006-2007 年信息系统建设优先发展的 9 个项目之一，为学科馆员提供网络化的个人空间，促进个人知识的组织化和隐性知识的显性化，进行业务协作，提交、发布业务信息，从而实现中国科学院国家科学图书馆学科馆员信息服务能力和管理水平的提升，更好支撑中

国科学院国家科学图书馆的服务创新。

2 系统设计

我们采用 B/S 结构来设计学科馆员工作平台，具体就是浏览器/中间层/服务器的“瘦客户机”模式。在这种模式中，客户端只需装上操作系统、网络协议软件、浏览器即可， HTTP 协议为所有的应用提供了统一的通信基础，所有的处理核心将在服务器端完成。

实现系统的 B/S 结构，我们选择的是 Struts 应用程序框架和 Hibernate 技术。Struts 是一种较好实现了 MVC 思想的技术框架，而且近年在业界开发中被证明是一种比较稳定、成熟的技术框架。以 Struts 为基础，可以使开发人员集中精力关注于构建业务应用程序，而不必关注体系结构上的问题。对于对象关系的映射和持久性数据的访问，采用了 DAO(Data Access Object)+Hibernate 的架构。先使用 DAO 模式来把数据访问封装起来，以供在其它应用层中统一调用。然后采用了 Hibernate 这样一种新的 ORM（Object Relational Mapping）映射工具，它同时提供从 Java 类到数据表的映射和数据查询、恢复等机制，实现数据层对象的持久性。Hibernate 可和多种 Web 服务器以及应用服务器良好集成,相对使用 JDBC 和 SQL 来手工操作数据库，使用 Hibernate 可以大大减少操作数据库的工作量。

学科馆员工作平台采用模块化设计和模块化开发，如图 1 所示，平台一共分为六大模块：工作管理模块、资源管理模块、团队管理模块，辅助功能模块、系统管理模块以及系统安全控制模块。各模块松散耦合，这使得平台中的任何一个模块的变化对系统其他模块的影响降至最低程度。对任何一个模块的理解、测试和修改，无须涉及系统的其他模块。各模块之间的通信通过数据库进行,任何一个模块都针对用户的不同角色设计了不同的功能。

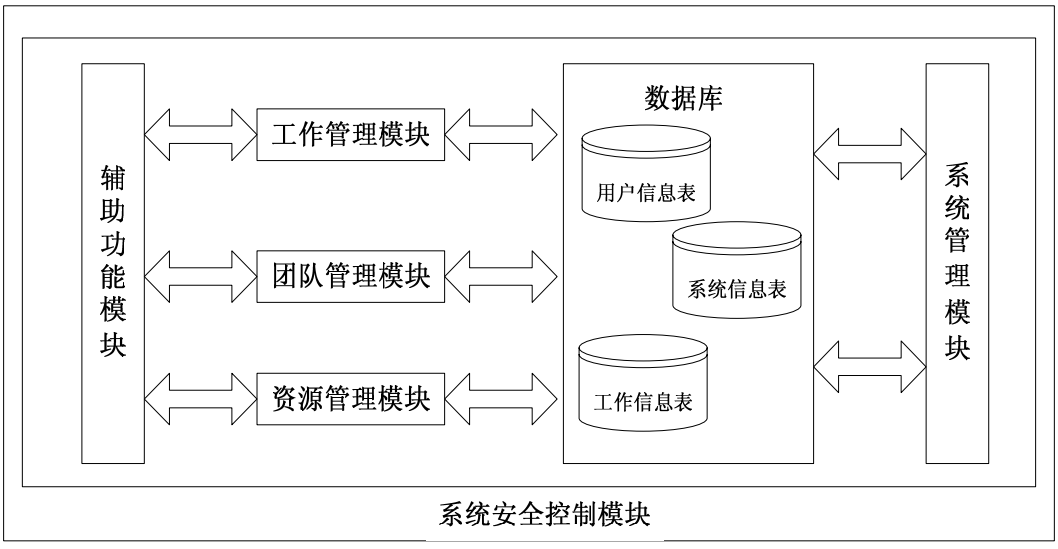

图 1 系统功能结构图

系统根据使用对象的不同将用户角色分成三种，普通学科馆员、学科馆员组长和系统管理员。

系统管理员负责整个系统的数据来源，拥有定义、管理用户和用户组，添加、删除、修

改栏目，管理文件的权限。普通用户则可以访问系统、浏览、查看、检索信息，上传下载文件，进行业务协作，管理工作计划日程。学科组长除了拥有普通学科馆员的一切权限之外，还可以查看该组学科馆员的工作计划，工作日程和工作总结，并向该组的学科馆员直接分配任务。

本系统的安全分为三个层次：数据层、服务器层、应用程序层。其中数据层与服务器层的安全控制由服务器软件与数据库软件实现，应用程序层则需平台自身实现。应用程序层的安全保证主要有：禁止未登录系统用户绕过登录模块访问平台资源；禁止恶意程序暴力破解用户密码。这个模块覆盖了平台内所有的资源，也就是说，没有通过系统安全控制模块权限验证的访问将被系统禁止。

3 系统实现

3．1 系统逻辑结构

从逻辑上讲，本系统在实现过程中遵循 J2EE 架构标准，如图 2 所示，分为四个层次，分别为表示层、业务逻辑层、持久层、数据层。

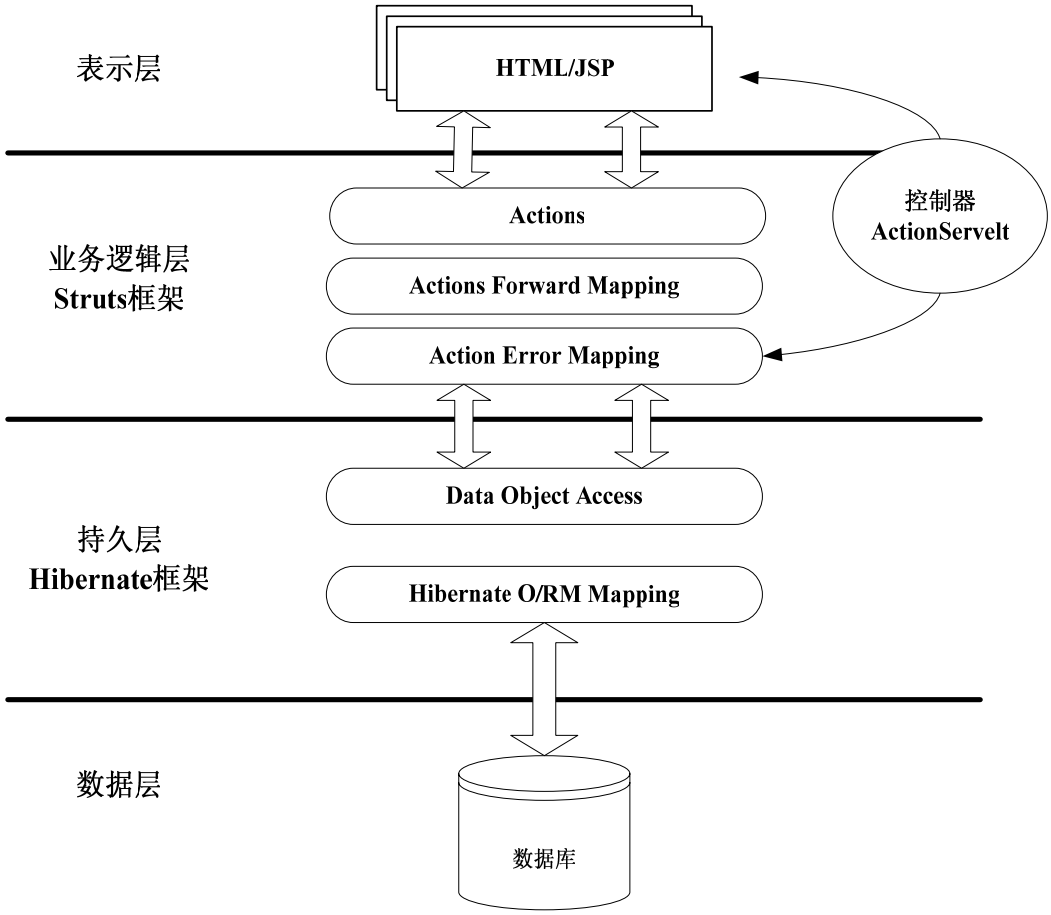

图 2 系统逻辑结构

表示层：管理用户的请求,做出相应的响应；提供一个控制器，将页面的请求委派给其他

层进行处理，为显示模块提供业务模型数据，提供 UI 界面的输入内容验证。

业务逻辑层：进行软件核心业务逻辑的处理。由若干运行在 Web 服务器下的 JavaBean 组成，这些组件是可以重复使用的，可以不做任何修改就移植到其他的服务器上。

持久层：建立持久化类及其属性与数据库中表及其字段的对应关系,提供简化 SQL 语句的机制，实现数据的 CRUD 操作(创建、读取、更新及删除)，数据库连接的建立与管理。

数据层：用于保存持久化数据。

通过把表示层、业务逻辑层和后端的数据服务分开，保持每层的相对独立性，松耦合度。每层只提供特定的服务，任何一层只要其提供的接口不变，发生变化时对其他层没有影响或者影响很小。使各层功能变得简单、专一、容易实现，而且具有更好的可维护性和可拓展性。

3．2 表示层的实现

表示层主要由 JSP 技术结合 Struts 提供的标签库 TagLib 来实现。JSP 技术仅用于显示用户界面, 发送 Struts 请求和接受响应完成交互, 实现显示和业务逻辑的解耦，页面可以完全不包含任何 java 代码, 这样 JSP 美工人员独立完成显示模块的开发。

Struts 框架的处理流程清楚地体现了模型（Model）——视图（View）——控制器（Controller）系统的特点。模型（Model）是应用程序的主体部分，表示业务数据或者业务逻辑。视图（View）是应用程序中用户界面相关的部分，是用户看到并与之交互的界面视图，主要由 JSP 建立。在 Struts 框架中控制器主要包括 Action Servlet 类和 Action 类, Action Servlet 是 Struts 的核心部件，它接受用户的 Http 请求，根据配置信息将请求转发给适当的 Action 对象，Action 类负责调用模型的方法，并帮助控制应用程序的流程； Struts 主要包括 web.xml 和 struts- config.xml 两个配置文件，其中 web.xml 是 Web 应用的发布描述文件，struts- config. xml 是与 Struts 相关的特殊信息配置的描述文件。

当用户从 login.jsp 页面发出登陆系统的 Action 请求后, ActionServlet 会通过查找 struts-config.xml 配置文件实现请求(*.do)到业务逻辑(Action)的映射，所有的业务逻辑封装在 Action 中，由 Action 调用业务逻辑的服务组件，根据处理结果跳转到 Forward 对象指定的响应。对应 struts- config.xml 配置如下：

```
<struts-config>
<! - - 登陆系统对应的 ActionForm- - >
<form-beans>
<form-bean name="loginForm" type="com.xkgy.form.loginForm" />
</form-beans>
<! - - 登陆系统对应的 ActionForm- - >
<action-mappings type="org.apache.struts.action.ActionMapping">
 <action input="/Login.jsp" name="loginForm" path="/loginAction" scope="request"
 type="com.xkgy.action.loginAction" />
 <forward name="Login" path="/Login.jsp" />
```

```
</action>
</action-mappings>
</struts-config>
```

3．3 业务逻辑层的实现

业务逻辑层负责封装数据持久层提供的数据对象，并为表现层提供功能接口。该层的数据来源于数据持久层的 Persistent Objects 和表现层的 FormBean，是表现层与数据持久层进行数据通信的中介层。由于在层次体系结构中责任的分离，因此这一层的重点在于设计对象的可重用性，保持与客户机的无关性。

在实现上对于业务逻辑层我们选择了普通的 Java 类来实现，而没有采用流行的 EJB。这是出于以下考虑：一方面，由于我们的应用是一个大集中的环境，不需要考虑分布式的应用，而普通 Java 对象的本地调用比 EJB 在内存和性能方面的开销要少得多；另一方面，普通 Java 对象特别适合于运行在服务器集群环境中。由于普通 Java 对象的所有实例都是等同的，因此对它的调用总能被路由到集群中性能最佳的服务器上。例如，updateUser 类处理用户信息修改的业务逻辑，其关键代码如下:

```
public void updateUser (EditUserForm form,int userid)
throws HibernateException {
Session session =this.beginTransaction();//初始化数据库连接事务
try {
     HumUserrinfo newuser=( HumUserinfo)session.load
     (HumUserinfo.class, new Integer (userid));//根据 edituser. jsp 页面中传来的用户 ID
     取得表中对应的用户信息的一条记录
       newuser.set (form.getDepartname ());//根据表单输入的用户部门名称重新设置用户
       所在部门名称
       session.update(newuser); //更新数据库表中的记录
       this.endTransaction(true);//完成一个数据库事务
     }
      catch (HibernateException e) {
       this.endTransaction(false); }
}
```

3．4 持久层的实现

在数据访问技术中，我们采用了 Hibernate 对象关系映射（ORM）框架。Hibernate 是一种比较彻底的 Java 对象映射工具，支持所有的使用各种 Java 思想（如 inheritance，association，composition，collections）等实现的对象。它可以直接映射大部分的 JavaBean 而不需要对它们作任何修改，即使修改也只是在 Bean 里面加上一些私有访问方法；可以将一个用户定义的多个类实例映射到一张表的同一行；可以利用代理模式简化载入类的过程。这

些都大大减少了利用 Hibernate QL 从数据库提取数据的代码编写量，从而节约开发时间和开发成本。此外 Hibernate 还具有平台无关性、性能高以及动态 Query 的特点。

系统的数据访问策略采用了 DAO 模式。通过 DAO 模式，为业务对象提供一个数据访问的抽象层，从而清晰地分离业务逻辑和数据访问逻辑。这样，当底层数据结构的发生变化时，可以不影响到应用系统的其它部分。DAO 模式的优势主要体现在：

（1）是面向接口的轻量级解决方案；

（2）能为业务组件提供数据库平台独立的调用；

（3）数据访问的接口由业务逻辑决定，持久层的变化不影响业务逻辑层的工作。

因此，系统中数据访问层的实现是通过定义一个 HibernateDAO 的基类，来实现 O-R Mapping。同时实现的 SessionManager 类为数据对象的访问提供 Session 的管理，包括创建一个 Session，得到已有的 Session 和关闭当前的 Session 功能。在每个 Session 中提供事务处理的功能。每个数据库表都会对应一个 Hibernate 映射文件，用于生成数据模型。例如用户信息的修改主要用到用户信息表 HumUserinfo(该表已在数据库中建好) 。在 eclipse 环境下, 建立 HumUserinfo 表的 Hibernate 映射文件 HumCourseinfo.hbm.xm，同时生成持久对象类 HumCourseinfo.java(可以由开发集成工具 Eclipse 自动生成，也可以手动编写, 主要由一些属性及对应的 getter/setter 方法组成)。为了让 Hibernate 知道怎么把一个对象存储到数据库中，需要在 hibernate.cfg.xml 中注册生成的映射文件，hibernate.cfg.xml 配置文件其关键代码如下：

```
<session- factory >
<! - - mapping files- - >
<mapping resource = "com/xkgy /pubmodel/HumUserinfo.hbm.xml" />
...
</session- factory>
```

4 结束语

本系统采用了J2EE 的多层开发框架, 并采用MVC模式分开了表现和逻辑，各层次之间是松散耦合,层与层之间的工作几乎是完全独立的, 而不同的业务逻辑基于不同的模块进行开发, 这样既利于系统业务的重用, 又利于团队的开发。其中Web层的设计在采用了Struts之后, 很好地把业务逻辑和表示层分离, 这种分离使得系统不至于“牵一发而动全身”, 同时, 也便于业务逻辑模块的重用。在处理数据持久性上, 采用Hibernate 来实现对象关系的映射, 以DAO模式脱离对特定数据源的依赖，把面向对象的设计开发与关系数据库联系起来, 优化了系统的数据层设计。

目前学科馆员工作平台一期工程已开发完成，已经在中国科学院国家科学图书馆推广使用。通过实际运行，服务器比较稳定, 同时经过不同终端的多客户的同时并发压力测试, 表明该平台在安全性、稳定性和健壮性上都有优良的表现，有效地支撑了国家科学图书馆的学科化信息服务。但是, 这个平台的原形实现只是一个完善系统的开始, 下一步我们准备引入Spring 模式, 进一步加强中间层的设计,强化元数据管理模块, 更好地实现以XML为基础的数据共享。

参考文献

[1]宋汉增，沈琳. 利用 Hibernate 对象持久化服务简化 Java 数据库访问. 计算机应用, 2003.

[2]雷钧，徐洪胜，付勇智. MVC 设计模式在 J2EE 平台上的应用.微计算机信息，2006,5- 3：1- 3.

[3]田珂，谢世波. J2EE 数据持久层的解决方案.计算机工程.2003 Vol.29 No.22

[4]Jreg Barish. J2EE Web 应用高级编程.北京清华大学出版社.2002.10

[5]孙卫琴.精通 Struts:基于 MVC 的 Java Web 设计与开发.电子工业出版社,2004

[6]飞思科技产品研发中心. JavaWeb服务应用开发详解. 北京:电子工业出版社, 2002.

[7] James Goodwill. Mastering Jakarta Struts. Wiley Publishing,Inc.2002

[8] The Struts User' s Guide [EB/OL].http:/ / Jakarta.apache.org

[9] http://www.hibernate.org/hib_docs/v3/reference/en/html/2005. 09.29

[10] http://www.jdon.com/designpatterns/index.htm 2005.08.12

作者简介：王峰，男，1979 年生，中国科学院国家科学图书馆武汉分馆，信息技术部，助理研究员，研究方向为数字图书馆技术，数据库技术。

联系电话：027-87199182，13659824025

E-Mail： wangf@mail.whlib.ac.cn

地　　址：湖北省武汉市武昌区小洪山西区 25 号中国科学院国家科学图书馆武汉分馆

邮　　编：430071

作者简介：梁慧刚，男，1977 年生，中国科学院国家科学图书馆武汉分馆，情报研究部，助理研究员，研究方向为学科情报研究。

联系电话：027-87199180

E-Mail:： lianghg@mail.whlib.ac.cn