应用 SpringMVC与 Hibernate进行 WEB开发

许文稼 赵英凯
(南京工业大学自动化学院 江苏 南京 210009)

摘 要 现在很流行在 Web应用系统设计中运用 MVC框架结构进行开发。在客户关系管理的 Web应用中,选用 SpringMVC作为系统构架模式。运用 Hibernate技术实现关系型数据库模型上的持久化,对 JDBC进行封装,提高 Model层的质量。运用 MyEclipse的 IDE组件进行基于 JDBC的数据库表的 OR映射工作。运用 DAO模式实现业务逻辑与数据逻辑分离,底层关系数据的结构变化不再对业务逻辑造成大的影响。

关键词 SpringMVC 对象持久化 Hibernate

APPLICATION OF SPRINGMVC AND HIBERNATE TO DEVELOPMENT OF WEB

Xu Wenjia Zhao Yingkai
(College of Automation, Nanjing University of Technology, Nanjing 210009, Jiangsu, China)

Abstract It is very popular to design the Web application system with MVC Framework. In the Web application of client relationship management, SpringMVC is chosen as the system framework. Hibernate technology is used to make the relational database model persistent. The JDBC is encapsulated, and the quality of the Model layer is improved. IDE component of MyEclipse is used in the object/relational persistence. DAO model is applied to separate the business logic and the data logic. The changes of the architecture of understratum relational data don't affect the business logic too much.

Keywords SpringMVC Object persistence Hibernate

0 引 言

目前进行 Web开发工作可选用的技术很多。早期的框架有 Structs,Webwork,如今又出现了 SpringMVC, Tapestry, JSF等。为开发客户关系管理,支持无线上网与有线上网,这里选用 SpringMVC进行开发,也称 Spring Framework。数据库选用 Mysql,使用 velocity做页面,由于 Spring有许多类用来支持其它的框架(如 Hibernate和 Struts),所以持久层的 Hibernate编码也得到了极大的简化。

1 SpringMVC框架简介

Spring是一种轻量级的容器,提供了一种新的机制来管理业务对象及其依赖关系。它的 MVC模型结构将对数据库的操作分为了界面视图部分,业务逻辑模型部分,数据库通信控制部分,对层次的分隔强、耦合低。

SpringMVC的最大特点是开发人员能通过视图与逻辑操作部分的接口 controller编程,使用 xml文件来简单地定义其实现。通过 ModelAndView handleRequest(request,response),使应用程序处理用户输入的表单 AbstractFormController,使多业务输入处理到一个表单 AbstractWizardController。

Controller的编程不但减少了耦合,更提供了方便的测试,可以对每一个 controller编写一个 test。

2 Hibernate持久化数据简介

对于数据层的持久化,我们使用 Spring的组件, ORM模板。Hibernate是 O/R映射框架,通过它就能将数据库中各个表的 contect与 JAVA中的对象一一对应,将数据映射成对象,对象映射成数据。由于在应用层和数据库之间建立了持久层,从而在编写 CRUD操作时只需注意 JAVA中的对象名称及属性。从另一方面, Hibernate是一种"对象—关系型数据",通过映射(mapping)". hbm. xml"文件将对象与关系型数据相关联。数据关联存在多种方式,如一对一关联,单向一对多关联,多对多关联,双向一对多关联,多对多关联。主要包括:

1) 初始化映射文件。这里是基于 MyEclipse平台进行操作的,所以直接使用 IDE就能从 MySQL数据库中导出表结构,自动生成对应的 ORM文件,将数据库的表映射成类, JAVA代码。

2) 建立 SessionFactory。SessionFactory负责 session实例, session负责操作持久化数据。

3) 利用 Hibernate封装了 JDBC连接。在 *. xml文件中配置 JDBC的属性,建立连接。

2.1 SpringMVC框架运行流程

按照图 1,图 2具体介绍 SpringMVC框架运行流程:

收稿日期: 2006—04—21。许文稼,硕士生,主研领域:控制理论与控制工程。

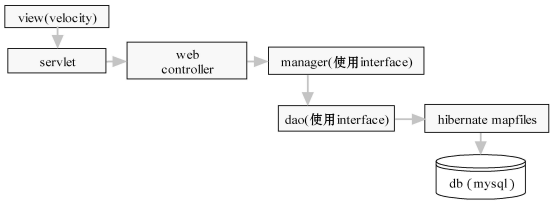

图 1　应用程序的文件结构图

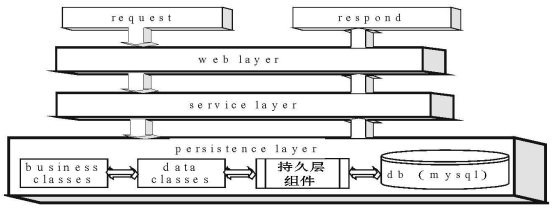

图 2　响应请求的实现层次图

1)将 Web页面中的输入元素分装为一个“请求”数据对象 form。本例中将登录界面中的用户名,密码封装为对象。

2)根据请求的不同,调度相应的逻辑处理单元,并将(请求)数据对象作为对象输入。

首先,这一步就是利用 SpringMVC中唯一负责调度的 DispatcherServlet将所有请求—HttpServletRequest中的输入数据配置分发至各个逻辑处理器。在 Web.xml中配置使 DispatcherServlet自动查找名为 action-servlet.xml文件。接着,将 action-servlet.xml的 mapping设置为“*.html”,这样 action-servlet就能处理所有以“.html”结尾的 URL。

其次,在“/users.html”后面,添加“UserController”这样 URL能够调用 UserController类。另外再建 UserLoginController处理编辑数据库的记录,在 action-servlet.xml中配置它,类似于 UserController。

```
<prop key="/users.html">userController</prop>
<prop key="/userLogin.html">userLoginController</prop>
```

为了使这些 Controller接口获取到数据列表,还需将它们依关系注射到 UserManager。Manager就是 Spring中的“业务委派”,很容易将数据层(DAO)和 Web层(servlets)分离,且对 Web层更为友好,可以接受 String类型,返回一个 User对象。通过在它的实现类 UserManagerImpl中设置私有变量 UserDAO,实现了 Web层与数据持久层的相互转化。可在里面添加需要的业务逻辑,如根据用户提交的 Email和密码验证用户登录信息。使用实现类 UserManagerImpl的好处是使用了 Inner-bean访问 userManagerImpl,这样外部不可以直接访问这个类,而必须要通过事物方式访问。

在 UserManagerImpl中添加 public UserDAO getUserDao(){ return userDao;}使得看不出在使用 Hibernate。这里联系到了 Hibernate的 DAO文件,将在 2.2中详细介绍。

最后,通过 Hibernate中的 UserDAO与底层数据库相连,实现对数据库的访问。

3)逻辑处理单元完成运算后返回一个结果数据对象。

4)将结果数据对象中的数据与预先设计的表现层相融合并展现给用户。将数据从逻辑业务层返回到 Web层,将输出页面返回给 Web服务器,再由 Web服务器返回给用户浏览器。

2.2　Hibernate与 SpringMVC的结合

2.2.1　与 Spring连接的配置

为了使 Spring找到 Hibernate,需要利用 ContextLoaderListener找到 Spring_HibernateDateAccess.xml,实现 Spring与 Hibernate的连接,还需在 web.xml中配置。

```
<param-name>contextConfigLocation</param-name>
<param-value>/WEB-INF/Spring_Hibernate_DataAccess.xml</param-value>
```

2.2.2　下面用 Hibernate实现 ORM持久层框架

1)初始化映射文件。它是基于 MYEclipse平台进行操作的,所以直接使用 IDE就能从 MySQL数据库中导出表结构,自动生成对应的 ORM文件,将数据库的表映射成类、JAVA代码。

通过 MYEclipse的插件,数据库中的每一张表格都分别映射生成三种文件。如与 UserStateinfo表格对应的文件有 AbstracUserStateinfo.java UserStateinfo.java UserStateinfo.hbm.xml三种。

实体映射文件 AbstracUserStateinfo.java指定与表格 UserStateinfo形成映射关系,表中的关键字段 stateId与变量 java.lang.Integer stateId呈映射关系。而 UserStateinfo.java主要作用是利用 super访问上级类 AbstracUserStateinfo中的变量。接着生成类表映射配置文件 UserStateinfo.hbm.xml。

```
<!-- DO NOT EDIT: This is a generated file that is synchronized -->
<hibernate-mapping package="com.contact.hibernate.mapfiles">
<class name="UserStateinfo" table="user_stateinfo">
<id name="stateId" column="State_ID" type="java.lang.Integer">
<property name="stateInfo" column="State_Info" type="java.lang.String"/>
<set name="userBasicinfoSet" inverse="true">
<key column="User_State"/><one-to-many class="UserBasicinfo"/>
```

根据上文总结出以下映射关系:

类表映射配置:映射类 UserStateinfo对应数据库表 user_stateinfo

Id映射配置:标示为 stateId的实体类对应数据库表中的主键字段 State_ID。

属性/字段映射配置:映射类中持久化的 JavaBean型属性 stateInfo对应库表字段 State_Info,由于表 UserStateinfo与表 userBasicinfoSet数据呈双向一对多的关联组合,所以需进行相关配置,首先是关联关系的控制方向允许反转,不再分主控方与被控方。这样“用户基本信息表”这一对象在持久化过程中,就可以主动获取其关联的“用户状态表”对象的 id。

接着配置 Hibernate在 Spring_HibernateDateAccess.xml中添加映像文件的路径,

```
<property name="mappingResources">
<list><value>com/contact/hibernate/mapfiles/UserStateinfo.hbm.xml
```

2)建立 SessionFactory。SessionFactory负责创建 Hibernate中 Session实例,这里采用线程的设计,可由多个线程并发调用。Spring中可以使用 TransactionManager进行 SessionFactory的配置解读,所以本文一直没用到 Hibernate.cfg.xml。而 Session是 Hibernate持久化操作的基础,贯穿了持久化管理器核心,提供了众多持久化方法。

3)利用 Hibernate封装了 JDBC连接。在 jdbc.properties文件中配置 JDBC的属性,建立连接。Hibernate.properties属于 Hibernate的 Configuration,负责管理 Hibernate的配置信息,取得底层实现的基本信息,其中配置了连接数据库的必要信息,如数据库类型名,数据库 URL,用户名及密码,数据库适配器。由于本文用的是 MySql数据库,所以这里用的数据库适配器是 org.hibernate.dialect.MySQLDialect。

(下转第 283页)

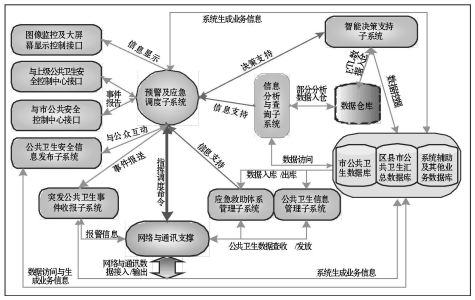

图 4　预警反应中心主要功能协作流程图

(2) 突发公共卫生事件业务流程设计

突发公共卫生事件处理包括突发事件收报、不明事件原因识别、事件应急处理(图 5)。公共卫生事件突然发生, 预警反应中心可通过人群、医疗机构、卫生行政部门等途径收到报警信息。对于已明原因事件, 预警反应中心接到报警后, 立即制定处理方案, 进行调度处理。对于不明原因事件, 除了按初步应急方案进行必要的应急处理外, 还应调度相关检测机构迅速查明原因, 根据结果调整完善初步应急方案, 进行相应处理。本级不能完全处理的事件, 向上级求援。

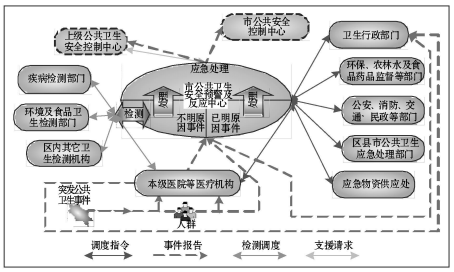

图 5　突发公共卫生事件业务流程图

(3) 非突发公共卫生事件业务流程设计

非突发公共卫生事件包括系统发现的异常事件原因识别、达到预警级别事件应急处理(图 6)。预警反应中心系统自分析发现异常, 指挥中心组织人员进行原因分析辨识, 必要时调度相关检测机构进行技术支持; 查明原因后, 达不到预警标准, 将不进行应急处理, 仅密切监视相关动态; 达到预警标准的, 按预警级别, 启动相应的预案, 按突发公共卫生事件应急处理程序调度处理。本级不能完全处理的事件, 向上级求援。

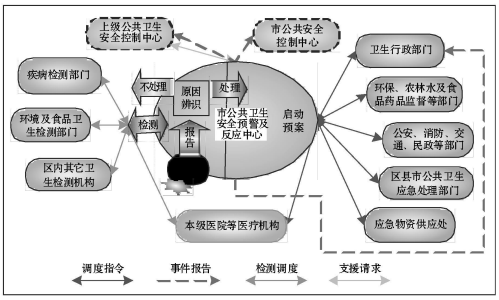

图 6　非突发公共卫生事件业务流程图

3　系统开发的技术重点分析

公共卫生安全预警反应系统在具体开发过程中, 应对部分技术进行重点研究。系统开发需重点解决的技术难点主要是公共卫生数据库的建设和信息分析与查询、智能决策支持两个子系统的开发。公共卫生数据库建设的核心就是公共卫生元数据标准的制定, 既要遵循国家标准, 又要与系统开发实际相结合。信息分析与查询子系统开发难点在于空间和非空间数据处理与统计, 可供选择的技术方法比较多, 但各有优缺点。智能决策技术近年来发展较快, 技术模型类别多, 适用范围各不相同, 智能决策支持子系统开发关键在于研究符合本系统运用的智能决策技术, 其中还涉及警情、警兆指标制定这一非常专业而复杂的工作。另外, 市四大基础数据库虽然不是本系统建设的内容, 但依赖市信息化建设的进度, 在公共卫生安全预警反应系统开发时, 基础数据库建设进度还存在较大的不确定性, 这时必须考虑建设相关的过渡数据库。

4　结　语

本研究对中小城市公共卫生安全预警反应系统进行了设计, 为其进一步详细设计和顺利开发奠定了良好基础。系统开发涉及面广, 技术难点较多, 具体任务应分步实施, 技术也不能一步到位, 难点需逐步解决。系统的技术问题和相应的体制矛盾也交织在一起, 系统开发时, 必须同步建立相应的配套体制, 系统才能正常运转。

参　考　文　献

[1] 高复先. 信息资源规划: 信息化建设基础工程[M]. 清华大学出版社, 2002.
[2] 鄢丹, 刘杰, 李洁. 食源性疾病预警与控制系统研究[J]. 计算机应用与软件, 2006, 23(9): 76-77, 95.
[3] 张永强, 刘泽军. 地理信息系统在公共卫生中的应用[J]. 中国公共卫生, 2005, 21(5): 632-633.
[4] 马家奇. 中国公共卫生信息化发展方向及策略探讨[J]. 中国公共卫生, 2002, 18(7): 895-896.
[5] 潘海东, 于明, 郑力. 全面构建国家公共卫生应急系统[J]. 办公自动化杂志, 2003(8): 14-20.

(上接第 265页)

4) 用 UserStateDAO实现持久层解耦。为了实现业务逻辑与数据逻辑相分离, 将数据访问划分为抽象层和实现层, 这里用了 DAO文件, 它为业务层提供了一个面向对象的接口, 对底层数据进行封装。文件中它继承了 Spring的 HibernateDaoSupport类, 可以用 getHibernateTemplate(), 来返回一个 HibernateTemplate对象, 获得 Hibernate DAO或是 SessionFactory。还利用了 Spring的 IoC(反转控制), 指定 UserStateDAO(Data Access Object,数据访问对象)类依赖 UserStateInfo类。

return (UserStateInfo) getHibernateTemplate(). load(UserStateInfo.class, ID);

3　结　论

基于 Myeclipse平台将 SpringFramework和 Hibernate相结合, 各取所长。成功地完成了客户关系管理的项目。经编写的 Hibernate_SpringTest程序测试, 用户登录之后可查看客户资料。

参　考　文　献

[1] 夏昕, 曹晓刚, 唐勇. 深入浅出 Hibernate[M]. 北京: 电子工业出版社, 2005.
[2] Matt Raible Spring Live[M/OL]. SourceBeat LLC, 2004
[3] CRAIG WALLS RYAN BREIDENBACH Spring in Action[M]. Manning Publications Co, 2005.