基于 Hibernate 与 Struts 的鱼雷库数据持久化研究*)

曾孝文[1] 侯楚林[2]

(湖南理工学院计算机与信息工程系 岳阳 414000)[1] (海军工程大学兵器工程系 武汉 430033)[2]

摘 要 整合 Struts 和 Hibernate 框架,设计了一个基于 MVC 模式的鱼雷仓库信息管理系统。文中对 Struts,Hibernate 的基本特征,以及二者的集成技术作了阐述,最后以该系统的检测仪器为例,对 Hibernate 实现系统持久化过程进行了详述。

关键词 关系映射框架, Struts,鱼雷,对象/关系映射,对象持久化

Data Persistence Research of Torpedo Store Information Management System Based on Hibernate and Struts

ZENG Xiao-wen[1] HOU Chu-lin[2]

(Department of Computer and Information Engineering, Hunan Institute of Science and Technology, Yueyang 414000, China)[1]

(Dept. of Weaponry Eng., Naval Univ. of Engineering, Wuhan 430033, China)[2]

Abstract Integrating both the framework of Struts and Hibernate, a torpedo store information management system based on MVC is designed, This paper narrates the basic character of two frameworks with Struts, Hibernate and the technology of combination of them. Finally, with the measuring instrument of this system for example, expatiate the implementation of hibernate persistence solution.

Keywords Hibernate, Struts, torpedo, O/R mapping, Object persistence

现代海战鱼雷种类和各种检测设备繁多,且更新快,故对鱼雷仓库管理提出了更高的要求。Struts 框架和 Hibernate 数据持久化技术,具有结构灵活、易于维护、扩展性好、数据访问效率高等优点,对实现鱼雷仓库信息的一体化、条理化、网络化建设具有极大的应用价值。

1 Hibernate 原理与 Struts 结构

Hibernate 是一种 ORM(Object-Relation Mapping 对象-关系映射)工具,能够建立面向对象的域模型和关系模型的映射[1]。其原理是应用程序通过 Hibernate 对数据库进行访问,对数据持久层操作,而 Hibernate 自身通过 properties 配置文件和 XML Mapping 映射文件将类映射到数据库的记录。从而 Application 应用可以通过模型中起着应用与数据库之间的桥梁作用的 PO 这个特殊的 Java Class 直接访问数据库,而不是必须使用 JDBC 和 SQL 进行数据的操作。经上述过程,通过 Hibernate 实现关系数据库的持久化操作。

Struts 是基于模型(Model)、视图(View)、控制器(Controller)的 MVC 模式应用架构,MVC 减弱了业务逻辑接口和数据接口之间的耦合,常被用来帮助开发者控制设计变化[2]。在 Struts 框架中,模型代表的是应用的业务逻辑,通过 JavaBean,EJB 组件实现;视图是应用的表示层,由 JSP 页面产生;控制器提供应用的处理过程控制。通过这种设计模型把应用逻辑、处理过程和显示逻辑分成不同的组件实现,组件之间可以进行交互和重用。

2 运用 Struts 和 Hibernate 开发雷库信息管理系统

2.1 系统的设计目的及构成

雷库信息管理系统的设计目的在于利用先进的计算机技术,构建一个网络化平台,实现装备由粗放型管理向集约型信息化管理转变,完成由过去单一鱼雷保障到多系列多品种鱼雷保障、由点式鱼雷补给到辖区内鱼雷补给网的任务,提高装备维护保养的能力。根据系统的业务流程和角色分析,可将系统划分为以下的功能模块:

(1)用户登录模块:提供人员登录、注册等的信息验证;

(2)人员管理模块:提供工作人员的年龄、军衔等基本信息的维护、查询;

(3)库房管理模块:提供库房编号、负责人等信息;

(4)装备管理模块:提供导弹、测试仪器、元器件、火工品、消耗品的型号、数量、相关责任人等信息;

(5) 出入库管理模块:提供人员和物资的出入库管理登记;

(6)性能状态管理模块:提供待用品、送修品、废品的信息。

2.2 系统的总体架构

由于雷库信息管理系统涉及到大量的数据处理与复杂的业务流程,我们使用 Struts 与 Hibernate 框架能很好地节约开发时间和开发成本。系统总体架构如图 1 所示,系统总体上可分为以下 5 个层次:

(1)客户层:运行在用户机器的浏览器中,处理与用户的交互。

(2) Web 层(视图层):运行在 Web 容器中,产生系统的表现逻辑,处理用户的请求并作出响应;整个 Web 层建立在 Struts 框架基础上,其中 View 由 JSP 页面组成;Controller 由 ActionServlet 结合 Struts-config. xml 和 Action 组成,而 Model 则交由业务逻辑层来实现。

(3)业务逻辑层:完成系统所需的业务,为 Web 层提供所

*)国家自然科学基金资助项目(Grant No. 50275146)。曾孝文 讲师,硕士研究生,研究方向为数据库、数据挖掘与信息系统、计算智能;侯楚林 博士生,研究方向为武器系统与运用工程。

需的业务方法，由 JavaBean 等 Business Objects(BO)构成。

(4)数据持久层：由 Hibernate 组成，Hibernate 从数据源中获取数据，然后生成持久对象(Persistent Objec，PO)，再把 PO 传给业务逻辑层。

(5)数据源层：即数据库层，存放雷库信息系统的数据。

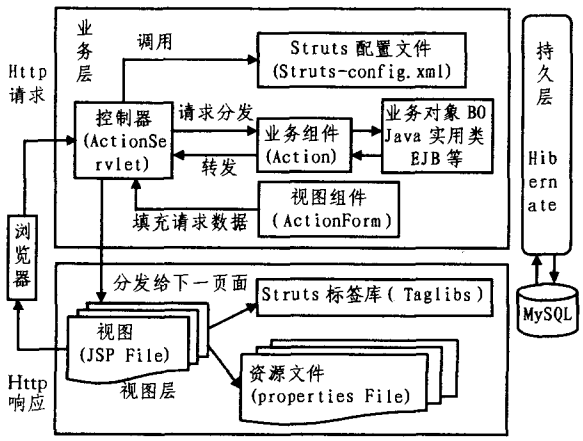

图 1　系统总体架构

3　数据持久层的实现

在系统开发过程中，使用功能强大、扩展性强的 Jbuilder2007 作为开发环境，；在数据库方面选择了易用性较好的 MySQL SERVER5.0，考虑到系统中各模块的实现大同小异，本文将以“装备管理功能模块”中的“测试仪器信息维护”为例说明系统数据持久化的实现。为了简化说明，在 Instrument 类中只定义了四个属性，即 InstruId，InstruName，InstrproductDate 和 torpedo，它的属性和数据库中 Instrument 表的字段是一一对应的，并且类型一致。

3.1　Hibernate 配置

Hibernate 从其配置文件中读取和数据库连接有关的信息，可以有两种格式，即 hibernate.properties 和 hibernate.cfg.xml[3]。因使用 XML 文件更加方便直观，故在本系统中，使用后者对 Hibernate 进行配置，配置文件 hibernate.cfg.xml 部分代码如下所示：

```
〈property name="dialect"〉
        net.sf.hibernate.dialect.MySQLDialect
〈/property〉
〈property name="connection.driver_class"〉
        com.mysql.jdbc.Driver
〈/property〉
〈property name="connection.url"〉
        jdbc:mysql://localhost:3306/sampledb
〈/property〉
〈mapping resource="mypack/ Instrument.hbm.xml" /〉
〈mapping resource="mypack/Order.hbm.xml" /〉
```

在以上属性配置文件中，定义了访问数据库的 URL 资源定位地址及数据库名称，以及符合 SQL 规范的 dialect 方言，在 mapping 映射文件配置部分，定义了 Instrument 表对应的映射文件 Instrument.hbm.xml，其他用到的映射资源可以随着系统开发进行灵活的加载与更新。

3.2　对持久化对象进行映射定义

数据库表对应的映射文件 Instrument.hbm.xml 包含了对象/关系映射所需的元数据。元数据中包含了持久化类的声明，以及类中各个属性到数据库表各个字段的映射关系[4]。映射文件 Instrument.hbm.Xml 主要代码如下所示：

```
〈hibernate-mapping〉
〈class name="mypack.equipment.Instrument" table="INSTRUMENT" lazy="true"〉
    〈meta attribute="class-scope"〉public〈/meta〉
    〈id name="InstrId" type="long" column="ID"〉
      〈meta attribute="scope-set"〉protected〈/meta〉
      〈generator class="native"/〉
    〈/id〉
  〈property name="InstrName" type="string"〉
      〈meta attribute="finder-method"〉findByName〈/meta〉
      〈column name="INSTR_NAME" length="15" not-null="true" unique="true" /〉
    〈/property〉
  〈property name="InstrproductDate" type="java.sql.Date"〉
      〈meta attribute="use-in-tostring"〉true〈/meta〉
      〈column name=" INSTR_PRODUCT_DATE " sql-type="DATE " /〉
    〈/property〉
  〈set name = " torpedo" table="INSTRUMENT_TORPEDO" cascade= "save-update"〉
    〈key column= "INSTRUMENT_ID"〉〈many-to-many class = "mypack.equipment.Torpedo" column= "TORPEDO_ID" /〉
  〈/set〉
    〈/class〉
〈/hibernate-mapping〉
```

在映射文件 Instrument.hbm.xm 中定义了 Instrument 类的为标识属性。通过定义 lazy="true"设置检索方式为延迟检索。映射文件中〈meta〉元素和〈column〉分别用于更加精粒度的描述类和表的定义。由于鱼雷的自导、控制、动力等部分分别有各自的测试仪器，测试仪器又可对多种雷测试，即它们可建立进行多对多关联映射。但在关系数据模型中，无法直接表达 INSTRUMENT 表和表 TORPEDO 之间的多对多关系，需创建连接表 INSTRUMENT_TORPEDO，它同时参照表 INSTRUMENT 和表 TORPEDO，以 INSTRUMENT_ID 和 TORPEDO_ID 为联合主键，字段 INSTRUMENT_ID 作为外键参照表 INSTRUMENT，而字段 TORPEDO_ID 作为外键参照表 TORPEDO。接下来的 many-to-many 则展示了 Hibernate 中实体之间多对多关系的定义方式，在此定义了 Instrument 类与 Torpedo 类的关联，即测试仪器与鱼雷的多对多关联。通过定义 cascade 属性设置级联保存和更新，即当保存或更新当前对象时，会级联保存和更新与它关联的对象。

3.3　定义持久化类

持久对象是一个完全符合 Java Bean 规范的纯 Java 对象，它包含有符合统一标准的属性和方法。其属性只可以通过自身的 get 和 set 方法访问，这样对外隐藏了内部实现的细节，规范了事务处理部分中每个属性所对应的数据库字段的数据操作。为简化开发本文采用在映射文件 Instrument.hbm.XmlHibernate 的元数据映射中添加〈meta〉元素，并结合 hbm2java 工具和 build.xml 文件自动生成 Instrument.java 文件，其生成的部分代码如下：

```
package mypack.equipment;
// Imports
public class Instrument {
  private int InstrId;
  public int getInstrId () { return InstrId;}
  public void setInstrId (int InstrId) {this.InstrId=InstrId;}
  private Set torpedo =new HashSet();
  public Set getTorpedo () { return torpedo;}
  public void setTorpedo (Set torpedo) {this. torpedo=torpedo;}
}
```

3.4　实现持久化操作

进行数据操作前先设置对应的 XML 配置文件 Hibernate.cfg.xml，接着执行以下操作即可完成数据库的连接，同时提供 hibernate 的对外接口：

```
Configuration cfg = new Configuration().configure
("/conf/hibernate.cfg.xml");
```

(下转第 273 页)

骤(n steps)。因此 M_J 可表示为:

$M_J = ACQUIRE_1, ACQUIRE_2, \cdots ACQUIRE_n$

应用 M_J 后需求的状态为:

$R_n(M_J) = ACQUIREn(\cdots(ACQUIRE2(ACQUIRE1(R_1, S_1, P_{j1}), P_{j2})\cdots), P_{jn})$

R_1 表示需求的初始状态,S_1 表示项目的初始环境,过程 M_J 每一步 step 依次采用的需求过程模式是 P_{j1}, P_{j2}, $\cdots$, P_{jn}。过程 M_J 的采用,必须明确满足相应的初始条件。如果过程说明在步骤 step i 需求工程师应该采用需求过程模式 P_i,那么过程环境条件 S_i 必须为真,而且需求的状态必须是 R_i。在人及问题两方面的复杂因素影响下,每一步的过程环境是不能预先判定 S_i 是否为真。

要定制目前已有的过程方法 M_J,例如 Robertsons 的 Volere 需求过程方法、Rational RUP 统一过程方法,详细描述了一个过程模型,包含一系列过程活动步骤 Step,每个活动步骤 Step i 描述了它的输入/输出,及其 Step i 推荐的技术。为满足软件组织开展当前项目的需要,需要为 M_J 进行定制时,可在 M_J 的每一步 Step i,遵循如下算法:

(1)检查需求的状态 R_i,包括已知需求信息和未知需求信息两方面。

(2)检查问题、方案及其项目特征,确定 S_i。

(3)判断建议该过程模型中推荐的模式 P_i 是否属于集合 $Agree(R_i, S_i, \chi(P))$,如果是将 P_i 作为过程的构造块进行复用;

(4)如果推荐采用的模式 P_i 不属于集合 $Agree(R_i, S_i, \chi(P))$,那么将当前模式 P_i 替换为 $Acknowledge(Evaluate(Agree(R_i, S_i, \chi(P)), AHP(P)), Sum(K, Wk, Ack))$。

如果为当前项目要创建一个新的过程模型,并且不考虑利用已有过程进行定制,那就需要重新定义一有效需求过程方法 M_J。可在需求过程模型 M_J 的每一步 Step i,遵循如下算法:

(1)检查需求的状态 R_i,包括已知需求信息和未知需求信息两方面。

(2)检查问题、方案及其项目特征,确定 S_i。

(3)根据需求工程师的个人情况,从可应用的模式列表中,选择合适的模式 $P_i = Acknowledge(Evaluate(Agree(R_i, S_i, \chi(P)), AHP(P)), Sum(K, Wk, Ack))$。

结束语 软件组织需要进行需求过程的规范化与改进,过程模式复用是软件过程改进的一条重要途径。本文提出了需求工程领域中过程模式的 FVRPEM 四视图的表示模型与复用准则。然而需求工程领域众多过程模式的存在,选择适用、可用的过程模式比较困难,为此给出了一种过程模式的复用方法,此方法融合需求过程模式库检索技术与基于准则的适用度评估方法,同时也将基于模式知识需求与需求工程师模式认知度作为模式选择的重要因素。

因此本文提出的基于模式的需求过程复用方法,是使需求过程的高层次重用成为可能。需求过程模式复用方法的推广,必将大大加快软件组织需求过程改进的步伐。

参考文献

[1] Pohl K. Process - Centered Requirements Engineering, Wiley, 1996

[2] Maiden N, Rugg G. ACRE: Selecting Methods for Requirements Acquisition. Software Engineering Journal, 1996, 11, (5): 183-192

[3] Wiegers K. Software Requirements, Microsoft Press, 1999

[4] Davis A. Software Requirements: Objects, Functions and States. Prentice Hall, 1993

[5] Playle G, Schroeder C. Software Requirements Elicitation: Problems, Tools, and Techniques. Crosstalk, 1996, 9(12): 19-24

[6] Gaska M, Gause D. An Approach for Cross-Discipline Requirements Engineering Process Patterns// Third International Conference on Requirements Engineering. IEEE Computer Society, 1998: 182-189

[7] Browne G, Rogich M. An Empirical Investigation of User Requirements Elicitation: Comparing the Effectiveness of Prompting Techniques. Journal of Management Information Systems, 2001, 17(4): 223-249

[8] International Workshop on Requirements Engineering Patterns - REP'04, September 2004 Kyoto, Japan, In conjunction with RE04, the 12th IEEE International Requirements Engineering Conference

[9] Hollenbach C, Frakes W. Software Process Reuse in industrial Setting, Florida//Fourth International Conference on Software Reuse Industrial. Orlando, IEEE Computer Society Press, Los Alamitos, CA, 1996

[10] Sommerville I, Sawyer P. Requirements Engineering. A Good Practice Guide, Wiley, 1997

(上接第 230 页)

```
SessionFactory sessionFactory = createSessionFactory (cfg);
```

得到了全局的 SessionFactory,就可以从中获取 session,进行对象的操作,如:save, update, load, delete, query 等。以查询条件仪器名称和生产日期,采用 HQL 检索方式为例说明查询方法,其相应的代码如下:

```
Query query = session. createQuery ("from Instrument as i where i. InstrName =: instrName "+" and i. InstrproductDate =: instrproductDate");
query. setString("instrName",自导测试仪);
query. setDate("instrproductDate",2006-8-8);
List results=query. list();
```

结束语 Struts 减弱了业务逻辑接口和数据接口之间的耦合,而 Hibernate 框架提供了数据持久层的支持,可以使开发人员专心地实现业务逻辑而不用分心于繁琐的数据库方面的逻辑,减小出错的机会。本文通过集成 Struts 和 Hibernate 框架技术,充分发挥了两者的优点,使得项目开发简洁、结构清晰,并且通过实例说明了利用 Hibernate 实现数据持久化的可行性。

参考文献

[1] 孙卫琴. 精通 HIBERNATE: Java 对象持久化技术详解[M]. 北京:电子工业出版社, 2005

[2] 孙卫琴. 精通 STRUTS:基于 MVC 的 JavaWeb 设计与开发[M]. 北京:电子工业出版社, 2004

[3] Hibernate 官方网站. [EB/OL]http://www. hibernate. org.

[4] 宋秀琴,侯殿昆,方中纯. 基于 Struts 和 Hibernate 的 Web 应用的构建[J]. 微计算机信息, 2005, 11(3): 125-127