第16卷 第11期
2006年11月

计算机技术与发展
COMPUTER TECHNOLOGY AND DEVELOPMENT

Vol. 16 No. 11
Nov. 2006

基于 Hibernate 和 Spring 框架的 Web 应用研究

华 涛[1,2],郝克刚[1],葛 玮[1]

(1. 西北大学 计算机科学系 软件工程研究所,陕西 西安 710069;
2. 聊城大学 网络信息中心,山东 聊城 252059)

摘 要:Hibernate是当前广泛应用的一种ORM数据库访问框架,它提供从对象模型到关系型数据模型的映射。Spring框架是一种轻量级J2EE应用框架,是针对Bean的生命周期进行管理的轻量级容器。介绍这两种框架并将它们结合起来形成一个业务持久层解决方案。这里提出使用Spring框架来管理业务对象,通过Hibernate作为数据持久层的方案来开发B/S架构的Web应用,并将此框架结构应用于实际项目的设计与开发中。

关键词:Hibernate;对象关系映射;Spring;反转控制

中图分类号:TP311.5 **文献标识码**:A **文章编号**:1673-629X(2006)11-0084-03

Web Research Based on Hibernate and Spring Frameworks

HUA Tao[1,2], HAO Ke-gang[1], GE Wei[1]

(1. Software Eng. Inst., Dept. of Computer Science, Northwest University, Xi'an 710069, China;
2. Network Information Center of Liaocheng University, Liaocheng 252059, China)

Abstract: Hibernate is a widely used, high performance ORM framework that provide the mapping between data model and relational data model. Spring framework is a light-level J2EE framework and a light-level container of managing beans' life recycle. In this paper, Hibernate and Spring framework is introduced in detail, and incorporate the two frameworks and form a resolution of business and persistence layer. Here Web application based on B/S framework is developed with Spring framework which manage business objects and Hibernate which is a resolution of data persistence layer. And apply the resolution to practice on the design and programming of project.

Key words: Hibernate; ORM; Spring; IOC

0 引 言

随着网上购物、电子商务和电子政务等Web应用越来越多,人们的工作效率得以提高。但是网络应用开发效率低、开发周期长等特点限制了网络应用的发展。J2EE的出现,逐渐改变了这一趋势而且J2EE也成为企业级网络应用程序事实的开发标准。但是,在J2EE应用程序的开发过程中也出现了很多问题:

1)随着面向对象技术和关系型数据库的广泛应用,它们之间的矛盾也逐渐突出:两者之间如何映射;在业务逻辑的开发过程中仍然使用SQL和JDBC技术进行数据库操作,编程过于复杂;当数据模型发生变化时,代码的变化量很大,降低了系统的可维护性和扩展性。

2)在J2EE应用程序中,企业级JavaBeans(EJB)构成了应用框架的基础。EJB要求应用系统的组件必须继承或依赖容器,使得单独测试EJB组件非常困难。以致使用EJB容器进行开发和调试需要耗费大量时间。

3)对于企业级应用开发,业务对象十分庞大而且对象之间关系错综复杂。所以,如何对它们进行良好管理也是一个问题。

文中提出使用一种ORM框架Hibernate,在Java对象和数据库表之间建立映射关系,以形成相对独立的对象持久层,从而降低程序和数据库的耦合度并简化程序开发。另外,使用一种轻量级J2EE框架Spring来管理对象持久层中的业务对象,使用松散耦合的方式集成分散的系统。

1 框架介绍

1.1 Hibernate框架

Hibernate是一种基于Java平台开放源代码的对象关系映射(ORM,Object/Relational Mapping)框架。它对JDBC进行了轻量级的封装,帮助开发者建立面向对象语言中的对象与关系型数据的之间的相互映射。它可以使人们脱离具体的数据库细节,简化JDBC编程,这样就可以实现程序的面向对象化和数据库的移植。并且开发者在设计阶段可以不必关心具体的数据库情况,完全使用面向对象思想来建立数据库模型[1]。

Hibernate不仅提供了从Java对象到数据表之间的映

收稿日期:2006-02-24

作者简介:华 涛(1980-),男,山东聊城人,硕士研究生,研究方向为软件工程;郝克刚,教授,博士生导师,研究方向为软件工程、软件理论、形式化方法;葛 玮,副教授,硕士生导师,研究方向为软件工程、软件测试。

射,也提供了数据查询和恢复机制。相对于使用 JDBC 和 SQL 来手动操作数据库,Hibernate 可以大大减少操作数据库的工作量。另外 Hibernate 利用装载模式简化了载入类的过程,减少了利用 Hibernate 本身提供的 HQL 语言从数据库提取数据的代码编写量,从而节约开发时间。针对当前流行的几乎所有数据库,Hibernate 都提供了相应的 Dialect 进行优化操作,提高了系统的效率。刚刚推出的 Hibernate3.0 又新增了大量的特性,如对存储过程和 XML 持久性的支持等[2,3]。

1.2 Spring 框架

Spring 框架是一种 Java 平台开放源代码的轻量级 J2EE 应用框架,它通过对反转控制模式(IOC)和面向方面编程(AOP)的实现解决了许多在以往 J2EE 应用开发中常见的问题。

IOC 模式又叫做依赖注入模式(Dependency Injection),是 Spring 框架的基础。通常应用代码在使用实例之前,需要创建对象实例。然而,IOC 将创建对象实例的任务交给 IOC 容器或者框架,使得应用代码可以直接使用实例。这样就由实现了 IOC 模式的 Spring 框架本身来判断业务对象彼此之间的依赖关系。通过使用 IOC,它提供了有效管理和组织业务对象的一致性方法并且鼓励了"对接口变成,而不是对类编程"的好习惯。并且 IOC 降低组件之间的耦合度,使得单元测试和集成测试更利于展开[4,5]。

面向对象编程主要是在系统的垂直切面关注问题,对于系统的横切面(系统级服务,比如日志、事务、安全性)关注较少。而 AOP 允许开发者动态修改 OOP 定义的静态模型,即可完成对横切面问题的解决。将这些系统级服务的抽象出来,并加以实现,使得程序中的重复代码能够大范围减少。

Spring 框架紧密地集成了 AOP 实现和 IOC 容器,从而解决了常见的应用问题。这种集成是以非侵入性的方式完成,也意味着可以根据具体项目情况选择其中某一个或多个模块来使用。

2 项目实例

这里通过一个企业级 Web 应用程序来简单介绍如何通过 Spring 和 Hibernate 这两个框架来整合开发系统。

2.1 系统架构

该项目是一电信企业效率管理系统。该应用需要对每天的各种业务模块进行处理分析,生成统计结果和一些系统管理模块操作,业务对象非常庞大,对系统的可靠性和安全性要求很高。若仍然使用 JDBC 和 SQL 来进行数据操作,代码量巨大,并且降低了系统的可维护性。业务对象及其关系的管理也需要单独的业务层进行处理。所以,当前的单一的基于 MVC 架构的开发框架并不适合项目要求,需要进行若干拓展。具体的系统框架如图 1 所示。

系统使用 Struts 框架、Hibernate 框架和 Spring 框架来进行架构。通过实现了 MVC 设计模式的 Struts 框架来完成表示层、页面导航和域对象的装配。在 view 层主要是使用 JSP 和 tag 来完成表示逻辑;控制层主要使用 Struts 框架提供的 ActionServlet 类和相应 XML 配置文件进行页面导航控制;Model 层主要是处理域对象的装配和调用相应的业务组件完成相应逻辑。对于 Struts 框架在此就不再详细介绍,这里重点关注业务层和持久层的架构实现。

每个业务组件都需要通过调用若干个业务对象来处理具体的业务逻辑。对于小型系统,业务对象之间关系简单,管理单个的对象并不困难,直接编写处理代码可以完成。但是,对于大型的企业级应用,业务对象繁多且它们之间的关系复杂,开发者通过 Spring 框架的 IOC 容器使得应用中的对象关系清晰、一致,而且一切对象都是可控的。通过 Spring 框架的 IOC 容器可以在创建持久化对象 Javabean 的相应 DAO 对象同时将 Hibernate 组件集成进来。这样在 DAO 对象中封装了使用 Hibernate 组件的数据库操作,并且对于它的生命周期可以由 Spring 框架来管理,这种架构方式划分了每层的责任,降低了组件之间的耦合度,减少了大量的数据库编码工作量。

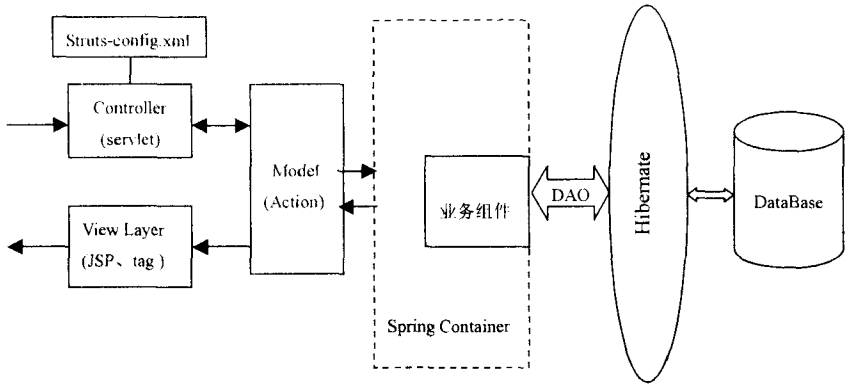

图 1　系统架构图

2.2 数据持久层实现

Hibernate 框架本质上是一个提供数据库服务的中间件。它是利用数据库以及其它一些配置文件如 hibernate.properties,XML Mapping 等来为应用提供数据持久化服务的。首先,需要创建持久化对象的 XML 配置文件,通过它 Hibernate 建立对象映射到数据库。这里,XML 配置文件需要注意各种关系的映射和参数配置,如 one to many,one to one,lazy 属性等,对系统的性能有很大影响。

另外 Hibernate Synchronizer 是一种免费的、能够同 Hibernate 框架配合生成代码的工具。当开发者修改完成 Hibernate 映射文件之后,它会自动生成修改后的持久化对象和相应的 DAO 对象,减少了手动编写的工作量。持久层实体和业务对象生成的类图如图 2 所示。

接口 HibernateDAO 定义了所有的数据库接口，而作为 DAO 的基类 HibernateDAOImpl 通过 Hibernate 的 API 实现了接口 HibernateDAO。对于具体持久化对象的 DAO 类来说，只要继承了 HibernateDAOImpl 类就能完成数据库操作。例如，保存或更新对象的方法为：

```
public void saveOrUpdate(Object obj) throws HibernateException {
  Transaction t = null;
  Session s = null;
  try {
    Session s = getSession();
    Transaction t = s.beginTransaction();
    saveOrUpdate(obj, s);
    t.commit();
  }catch (HibernateException e) {
    if(null ! =t) t.rollback();
    throw e;
  }
  finally {
    if (null ! =s) s.close();
  }
}
```

上面的语句作为方法封装在 HibernateDAOImpl 类中。另外从上面代码中可以看出，与使用 JDBC 和 SQL 编程相比，Hibernate 的代码量更少，条例更清晰，最重要的是脱离了具体数据库细节，修改非常方便。

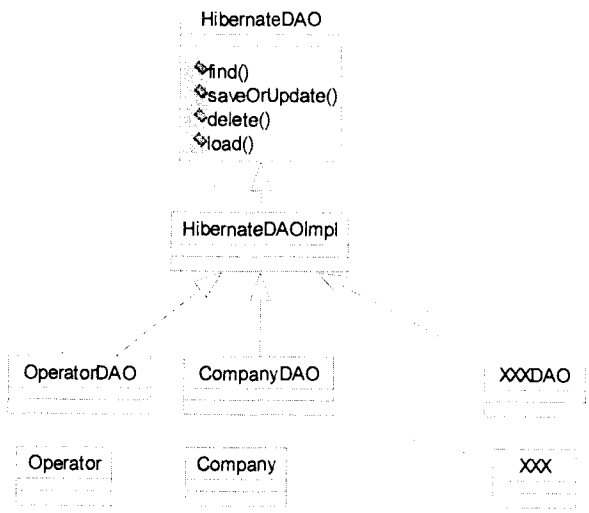

图 2 持久层类图

2.3 业务层集成

有了对象及其 DAO，接下来需要考虑如何配置业务对象，利用它们来执行 Model 层请求的逻辑部分，调用持久层进行数据库操作。为了良好的配置和管理业务对象及其关系，选择使用实现了 IOC 和 AOP 的 Spring 框架是一种比较好的策略。

这里通过图 2 中 Operator 对象来说明其集成方式。例子中 Operator 其对应的相关 Spring 配置文件如下：

```
<bean id = "HibernateUtil" class = "cn. com. cudes. po. util. HibernateUtil"/>
<bean id = "operatorDAO" class = "cn. com. cudes. po. entity. dao. OperatorDAO">
  <property name = "sessionFinder">
    <ref bean = "HibernateUtil"/>
  </property>
</bean>
```

Spring 框架是通过 setter 方法来注入依赖性的。可以看出在 Operator 的 DAO 类中肯定存在一个 setSessionFinder 方法，它本质上是一个 Hibernate Session 对象的设置方法。Hibernate 只有通过 Session 对象才能支持事务处理。当使用 Spring 创建 OperatorDAO 的实例时，Spring 的 IOC 容器会自动查找对应的 HibernateUtil 实例，若不存在，则创建其实例，通过 setter 方法传递给 sessionFinder 属性。Spring 通过 IOC 容器把 DAO 对象和封装了 Hibernate session 对象的 HibernateUtil 自动搭配起来，对象之间的关系处理全部交于它处理。

这只是一个应用 Spring 框架的简单实例。当然，可以定义对象的业务组件 bean，将对象的 DAO 通过 Spring 容器与其组装起来，以完成与该对象有关的业务逻辑。由于 Spring 采用了大量的 Java 反射机制，以非侵略性方式来设计对象，使得业务组件更加利于测试。它可以避免使用 EJB 重量级容器的种种缺点，如开发和测试困难等。

3 结束语

介绍了 Hibernate 和 Spring 框架的特点，详细描述了如何整合这两种框架来构建 Web 应用，解决了 J2EE 应用程序开发中常见的问题，大大减轻了编码的工作量。该实现方案已经在一个企业级系统中成功应用。

参考文献：

[1] Hibernate Reference Documentation[EB/OL]. 2005 - 04. http://www. hibernate. org/hib_ docs/v3/reference/en/html/.

[2] 董洪衫，窦延平. 利用 Hibernate 的 J2EE 数据持久层的解决方案[J]. 计算机工程，2004，30：17 - 19.

[3] 田 珂，谢世波，方 马. J2EE 数据持久层的解决方案[J]. 计算机工程，2003，29：93 - 95.

[4] Johnson R. Introducing the Spring Framework[EB/OL]. 2003 - 10. http://www. theserverside. com/ariticles/article. tss?l = Spring Framework.

[5] 罗时飞. 精通 Spring[M]. 北京：电子工业出版社，2005.