文章编号:1674-7070(2010)05-0455-06

基于 SSH + DWR 的 Web 开发框架研究与应用

陈遥[1]　李珊[2]　赵英男[1]

摘要

使用 Web 框架能有效提高 Web 程序的开发效率、增强程序的可维护性和可扩展性等. 结合分层设计的思想和目前一些流行开发框架的特点,给出一种基于 SSH + DWR 的新型 Web 框架设计与研究,以面向接口编程方式设计程序的各层,以 Struts 将表示层与业务层解耦,以 Hibernate 应用 DAO 模式将业务层与持久层解耦,以 Spring 的 IOC 达到组件之间的松耦合性,还以 DWR 改善用户界面响应不足,从而使程序具有扩展性强、重用性高和界面相应灵敏等性能,并以实例进行分析说明.

关键词

SSH;DWR;Web 框架

中图分类号 TP311.5

文献标志码 A

0　引言

Introduction

随着 Internet 覆盖范围的不断扩大,围绕 Web 应用系统的开发竞争日趋激烈,企业则需缩短产品开发周期、提高产品质量、降低成本和改进性能,来定制具有个性化的产品去占领市场以赢得竞争,故 Web 框架技术应运而生,因为框架是整个系统或系统的一部分的可重用设计,由一组抽象的类及其实例间的相互作用方式组成[1],具有一定的可重用性、可扩展性、成熟的稳定性,还可以处理系统很多细节问题,所以应用框架开发系统极大地提高了 Web 应用系统的开发质量和缩短开发周期等.

随着框架技术的发展以及各种框架的推陈出新,结合其中一些框架技术的特点,根据松耦合性、可扩展性、可重用性和可维护性的软件设计目标,本文将对一个新型 Web 系统开发框架,即一种基于 SSH(Struts + Spring + Hibernate) + DWR(Direct Web Remoting)的 Web 开发框架进行研究、改进与应用. 下面内容主要包括首先介绍了该框架的算法,然后给出了其应用实现,最后对整个工作进行了总结.

1　基于 SSH + DWR 的 Web 框架设计

Design of web framework based on SSH + DWR

从框架技术的目的出发,本文构建 Web 框架所采用的主要思想是:首先,对整个 Web 应用程序结构采用分层结构进行处理,具体而言,分层按系统的职能进行划分,将不同操作和处理置于不同层中进行处理,并且在开发过程中确保各层之间避免不必要的关联,层之间的通信只能通过定义的接口来执行,所有下层向上层提供调用的接口,具体实现细节对上层透明,这样可以降低系统各层之间的耦合程度,从而具有高度可扩展性、可重用性和可维护性;其次,在此基础上改善其中某些方面的不足(如客户端响应缓慢等).

根据上述构建 Web 框架的思想,结合目前比较流行的一种基于 MVC 模式传统 Web 框架 SSH[3] 和一种基于 Ajax 技术的框架 DWR[4] 的优点,整合出一个具有一定集成度的软件开发架构即一种基于 SSH + DWR 的新型 Web 框架(图 1),延伸出一些新的特性.

首先,将 Web 应用程序总体框架按照层次结构,从职能上为 3 个层次.

收稿日期 2010-09-05

资助项目 国家自然科学基金(60702076)

作者简介

陈遥,女,讲师,博士生,主要研究方向为计算机应用技术与数据挖掘. chenyao0077@163.com

1　南京信息工程大学　计算机与软件学院,南京,210044

2　南京航空航天大学　经济管理学院,南京,210016

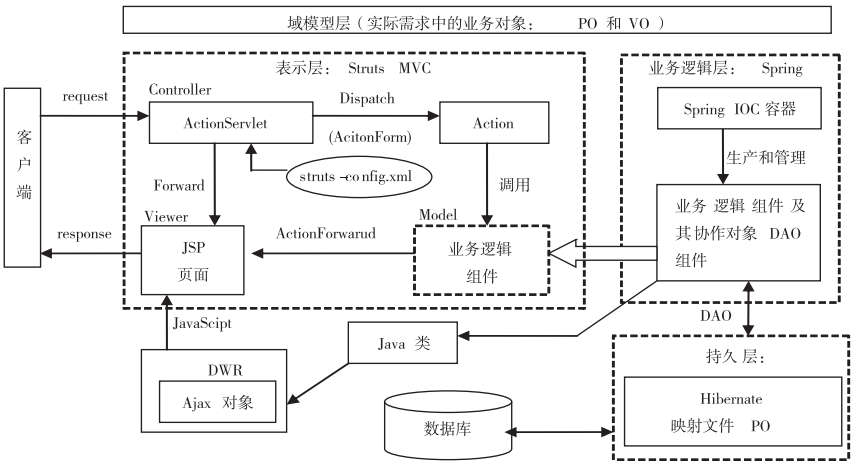

图 1 基于 SSH + DWR 的 Web 框架

Fig. 1 Web framework based on SSH + DWR

1）表示层（最上层），负责页面显示，响应用户请求，调用其下层的业务逻辑组件，并将结果返回到客户端. 该层主要使用 Struts 框架实现，Struts 主要采用 Servlet 和 JSP 技术实现 MVC（Model-Viewer-Controller，模型-视图-控制器）模式，很好地使显示、控制和模型相分离，即由控制器（负责信息处理和对业务逻辑的调用）将表现逻辑（表示层）和业务逻辑（业务层）解耦，实现了分层结构. Struts 通过其配置文件 struts-config. xml 部署整个系统的业务流程（图 1）. 其基本工作流程是：所有客户端请求都被提交给控制器 ActionServlet 处理，控制器将根据其配置文件决定是否要调用业务逻辑控制器 Action，若不要则直接转向请求的 JSP 页面，否则将请求转发给相应的 Action 对象处理并根据处理结果（ActionForward 对象）跳转到 Forword 对象指定的响应页面，其中 Action 对象会根据 ActionForm（封装用户请求参数）处理用户请求，并调用 Model（即业务层）的业务逻辑组件的方法来完成业务功能.

2）持久层（最底层），负责数据的存取、数据库的备份和同步等. 大框架通过数据访问对象（DAO）封装对数据库的数据访问细节，上层所有的数据库访问都通过调用 DAO 对象完成. DAO 对象通常包括：对持久化类（即 PO）的基本 CRUD 操作（插入、查询、更新和删除）. 主要采用 Hibernate 框架实现持久层，Hibernate 通过其对象-关系映射文件（ *. hbm. xml）将对象（指 PO）和数据库表相关联起来，并将 PO 中的属性和数据库表的字段一一对应，然后通过操作 PO 可以对数据库表中的数据进行基本 CRUD 操作，达到通过操作对象实现操作数据库的目的，代替了传统的通过 JDBC 和 SQL 语句对数据访问，简化了数据访问的复杂程度.

3）业务层（中间层），是系统的核心层，需要由业务服务对象来执行应用逻辑、获得从用户接口层的请求、执行向持久层的调用和处理事务等功能. 大框架中，Spring 使用其核心机制即“控制反转 IOC（或称依赖注入 DI，指在运行期由 Spring IOC 容器根据其配置文件将组件之间的依赖关系注入到组件中）”把各个对象以松耦合的方式组织在一起，而且各层对象的调用完全面向接口. 其中，Spring IOC 容器是产生 Bean（即组件或对象）的工厂，可以管理对象及对象间关系，可以通过编写其配置文件来设置对象关系和初始值. 这样容器在启动之后，所有对象都直接可以使用，不用编码来产生对象，所以 DI 机制实现了组件的即插即用. 因此，在业务层中，由管理组件的 Spring IOC 容器负责向 Action 提供业务模型（Model）组件及其协作对象 DAO 组件来完成业务逻辑，并提供缓冲池、事务处理等容器组件以提升系统性能和保证数据的完整性.

其次，上述层次结构设计 Struts + Spring + Hibernate（SSH）为传统 Web 应用框架，它的用户界面对用户自然和响应灵敏方面有所欠缺，因为传统 Web 应用程序使用“请求→响应”同步模式：每次应用的交互都需要由用户向服务器发送请求并空闲等待（白屏），直到从服务器返回新的 Web 页到客户端. 此种页面的刷新完全依赖于从服务器重新载入页面，从而导致了用户界面的响应灵敏度不高. 故出现

了一种 Ajax(Asynchronous JavaScript and XML)技术,其主要思想是相当于在客户端和服务器之间加入一个 Ajax 引擎,它允许客户端与服务器以异步方式交互,这样用户不用打开空白窗口等待服务器响应,而可继续客户端的其它工作;另外,服务器响应完毕后,将结果提交给 Ajax 引擎,Ajax 引擎使用 JavaScript 和 CSS 来相应地更新用户界面,而不是刷新整个页面,所以 Web 站点看起来是即时响应的. 因此本框架在上述 SSH 的表示层中加入了一种 Ajax 开源框架 DWR,DWR 通过 dwr.xml 配置要向客户端公开的服务器上某些服务(Java 类),运行时根据 Java 类动态生成 JavaScript 对象,以便 Web 页面能使用这些对象来调用该服务,就像它们是直接使用服务一样,从而构建出了一种更为动态和响应更灵敏的新型 Web 框架(图 1).

2 应用实例
Example of application

网上商店管理系统,一般分为前台模块和后台管理模块,由于系统的模块较多,本文主要以该系统的后台管理模块中的一个用户管理子模块为例说明该种基于 SSH + DWR 的新型 Web 框架的应用.

首先确定应用程序的业务实体,在分析业务实体的过程中,根据需要创建一域模型,然后实现域模型,为每个实体创建一个持久层的 Java 对象(PO),如本例模块中涉及一个 PO:用户对象 User.java(用户账号、用户名、密码和积分等信息). 其中用 JavaBean 的格式为上述对象的字段设置 setter 和 getter 方法.

2.1 数据持久层

Hibernate 通过映射文件将对象与数据库关联起来,仅需为每个 PO 编写一个映射文件(*.hbm.xml). 例如,用户对象 User.java 的映射文件为 User.hbm.xml. 在上述 PO 的基础上,设计数据访问对象 DAO 接口及其实现. Spring 对 Hibernate 的 DAO 实现提供良好的支持,让它继承 Spring 中的工具类 HibernateDaoSupport,这样可以使用该类提供的访问数据库的方法. 例如,用户管理接口 UserDao 及其实现 UserDaoImpl 的涉及片段如下:

```
public interface UserDao{……//定义对 PO 对象 User 的 CRUD 操作方法}
public class UserDaoImpl extends HibernateDaoSupport implements UserDao {
……//其接口 UserDao 中所有方法的实现
}
```

2.2 业务层

在大框架中,Spring 除实现业务逻辑的功能外,还扮演着核心框架作用:负责将 Struts、DWR、Spring 和 Hibernate 整合到一起,即 Spring 主要通过 IOC 根据其配置文件产生相关对象以实现关联关系;同时采用编程到接口,这样把各层组件以松耦合方式连起来. 以用户管理中业务为例,通过业务建模创建用户服务对象接口 IUserService 及实现 UserServiceImpl,封装对用户的权限管理等功能,并通过调用数据访问类 UserDao 实现对用户数据的操作. 其中重要的是,通过对 applicationContext.xml 的配置(具体如下文),将业务层所定义的接口和实现与 DAO 所定义的接口和实现连接起来,并为它们提供事务管理.

1) 关联 Hibernate

采用 Spring 的组件提供的数据源来实现数据库连接,定义数据源涉及到的配置片段:

```
<bean id = "dataSource" class = "org.springframework.jdbc.datasource.DriverManagerDataSource" >
<! --下面指定数据库驱动、数据库 URL、用户名、密码等参数-- >
</bean >
```

DAO 的配置必须依赖于 SessionFactory,因所有线程都是从 SessionFactory(主要负责创建 Session 对象)中获取 Session 对象(主要功能是操作映射的实体对象的实例)操作数据库. 定义 Hibernate 的 SessionFactory 配置如下(mappingResources 属性包含了映射文件的路径,list 下可配置多个映射文件,关联 Hibernate 配置):

```
<bean id = "sessionFactory" class = "org.springframework.orm.hibernate3.LocalSessionFactoryBean" >
<property name = "dataSource" > <ref local = "dataSource"/ > </property > <! --注入 id 为 dataSource 数据源-- >
<property name = "mappingResources" >
<list > <! --以下用来列出所有的 PO 映射文件-- >
<value > User.hbm.xml </value > …
</list >
</property >
</bean >
```

2) 构建事务管理组件

事务管理器的实现类有多种,大多数情况下,在业务层使用 Spring 的声明式事务管理,为目标对象(需要事务的业务对象 service 和 DAO 对象等)提供事务增强,具体如下:先把 service 和 DAO 注入到

Spring 容器,并需要注入一个 Hibernate 的局部事务管理器,最后通过 TransactionProxyFactoryBean 为每个目标对象配置一个代理. 以配置 UserService 声明式事务管理组件为例说明事务管理配置(其中,由 target 指定需要被代理的目标对象,并通过 transactionAttributes 属性,可以指定事务的管理策略):

```
<! --配置持久化类的 DAO 组件-- >
< bean id = " UserDao" class = " com. wind. web. DaoImpl. backManage. UserDaoImpl" >
< property name = " sessionFactory" > < ref local = " sessionFactory"/ > </property >
</bean >
<! --需要被增强的 bean 通常被命名为 xxxTarget-- >
< bean id = " UserServiceTarget" class = " com. wind. web. services. UserServiceImpl" >
< property name = " UserDAO" > < ref local = " UserDAO"/ > </property > <! --注入 id 为 UserDao 的引用-- >
</bean >
< bean id = " transactionManager" <! --配置 Hibernate 的局部事务管理器-- >
class = " org. springframework. orm. hibernate2. HibernateTransactionManager“ >
< property name = " sessionFactory" > < ref local = " sessionFactory"/ > </property >
</bean >
<! --声明式事务管理器,被代理之后的 service 具有事务功能,程序中就使用它-- >
< bean id = " UserService" class = " org. springframework. transaction. interceptor. TransactionProxyFactoryBean" >
< property name = " transactionManager" > < ref bean = " transactionManager"/ > </property >
< property name = " target" > < ref local = " UserServiceTarget "/ > </property > <! --需要被代理的目标-- >
< property name = " transactionAttributes" >
< props > <! --这里指定事务的管理策略-- > </ props >
</property >
</bean >
```

3) 关联 Struts

关联 Struts,采用 Spring 推荐的整合策略:将 Struts 的 Action 配置成 DelegatingActionProxy(继承于 Struts 的 Action),目的是将用户请求转发给 Spring 管理的 Bean,即所有的请求先被 ActionServlet 截获,请求被转发到对应的 Action,而 Action 的实现类全都是 DelegatingActionProxy,它再将请求转发给 Spring 容器管理的 Action. 其具体实现见 2.3.

2.3 表示层

2.3.1 Struts 框架

用 JSP 和 Struts 的 TagLib 库处理显示功能,表示层利用 ActionServlet 将请求(* . do)映射到相应的 Action,并由 Action 调用 Spring 管理的业务逻辑的服务组件,然后根据处理结果跳转到 Forword 对象指定的响应页面. 下面以一个后台管理页面上提交用户登录信息验证为例,设该 JSP 页面上实现该功能的跳转路径为:http: // …/backManage. do? method = backLogin. 实现该功能主要涉及 3 个方面.

1) 在 struts-config. xml 中的配置. 配置从特定的请求路径到相应的 Action 类的映射,此处必须配置 Action 的 type 元素为 DelegatingActionProxy,以调用 Spring 管理的 Bean. 配置关联 backManage. do 的 Action 如下:

```
< action-mappings >
< action path = "/backManage" <! --指定访问 Action 的路径-- >
scope = " request"
parameter = " method" <! --根据页面上跳转路径的 method 跳转到 Action 中具体方法-- >
type = " org. springframework. web. struts. DelegatingActionProxy" >
< forward name = " succeed" path = " LoginSucceed. jsp"/ >
</action >
</action-mappings >
```

2) 整合 Struts 与 Spring. applicationContext. xml 的配置片段如下,其中关联到 Spring bean 的 name 值要和 struts-config. xml 中的 path 值一致,这样当 Struts 加载对应的 Action 时,DelegatingActionProxy 就根据传入的 path 属性,在 Spring Context 寻找对应 bean,并将其实例返回给 Struts.

```
< bean name = "/backManage" class = " com. wind. web. backManage. BackManageAction" >
<! --注入上面定义的 id 为 UserService 的引用-- >
< property name = " UserService " > < ref bean = " UserService "/ > </property >
</bean >
```

3) 用户的 Action 实现类. 负责真实的处理(如 BackManageAction),并通过在 struts-config. xml 中配置 Action 的属性 parameter = " method" 传递给 Action,DispatchActionProxy 然后会从请求中获取该字段的值,并调用相应的方法,例如上述请求中通过 method 的使用调用了 Spring 组件 BackManageAction

中的 backLogin 方法.

```
public class BackManageAction extends DispatchAction{…
//后台操作方法的实现}
```

2.3.2 DWR 框架

DWR 是用来改善用户界面的灵敏度等作用的.下面以使用 DWR 实现验证系统中数据功能为例,即在表示层的 JSP 页面中使用该技术对数据进行相关格式验证,确保不会有误操作,特别是避免页面的跳转,加强页面的人性化.

1) 在文件 web.xml 中,添加 DWRServlet 的映射来加载 DWR 框架(略).

2) 配置 DWR。添加 dwr.xml 文件,其 create 元素告诉 DWR 应当公开给 Ajax 请求的服务器端类(由节点 param 指定的 JavaBean),使得 DWR 在运行时给这些 JavaBean 生成相应的 javascript 库(即对后台 JavaBean 调用的封装),这样 JSP 页面中可以直接使用该库来实现直接调用 Java 类的目的.下面配置做数据验证工作的 TestDao(由 Spring 创建的对象)的 javascript 库(库名为 Test,由节点 create 的属性 javascript 设置,这也是 JSP 页面上调用该对象时所用名称),实现 JSP 页面中通过 Test 来调用 TestDao 中方法,其 dwr.xml 主要配置片段:

```
<! --creator = "spring" 表示使用 Spring 创建实例,dwr 整合 Spring-- >
<create creator = "spring" javascript = "Test" >
<! --value 的值为定义在 applicationContext.xml 中某个 bean 的 id 值-- >
<param name = "beanName" value = "TestDao" > </param >
</create >
```

其在 Spring 的 applicationContext.xml 中对应部分:

```
<bean id = "TestDao" class = "com.wind.web.ServiceDaoImpl.TestDaoImpl" >
<property name = "sessionFactory" > < ref local = "sessionFactory"/ > </property >
</bean >
```

3) 执行验证过程中,JSP 页面上用 JavaScript 程序(运行于浏览器上,如 init()方法)来调用 DWR 框架为它配置的对应 Java 程序(运行于服务器上,如 init2()方法),其主要代码片段如下(其中,必须载入 DWR 中的核心 javascript 库 engine.js、util.js 和 DWR 运行时动态生成的 javascript 库(如 Test.js)):

```
<script type = 'text/Javascript' src = '/SSHD/DWR/interface/Test.js'> </script >
<script type = "text/Javascript" >
function init(s1){Test.init2(s1,getAnswer);//JavaScript 调用后台方法 }
function getAnswer(data){…//验证结果的提示 }
</script >
上面对应的数据处理的 Java 程序
public class TestDaoImpl implements TestDao {
public String init2(String s) {…//验证}
}
```

3 结束语

Conclusion

本文围绕目前一些流行开发框架和分层设计的思想,给出了一种基于 SSH + DWR 的新型 Web 框架的设计及其实现.该框架主要特色在于,通过 Struts 应用 MVC 模式使表示层与业务层解耦,并通过 Hibernate 应用 DAO 模式使业务层与持久层解耦,同时对各层及之间的设计采用面向接口编程方式,并通过 Spring 的 IOC 方式根据 Spring 配置文件在运行时实现各层对象的创建和关系注入,达到组件之间的松耦合性,从而使该框架具有扩展性强、重用性高、可维护性良好等性能;另外,通过 DWR 改善了传统框架中用户界面的响应灵敏度等不足.除了上述特点之外,该框架还可在提高系统安全性和易于使用等方面进一步研究.

参考文献

References

[1] Gamma E,Helm R,Johnson R,et al. Design patterns:Elements of reusable object-oriented software[M]. Addison Wesley,1994

[2] 李刚. 整合 Struts + Hibernate + Spring 应用开发详解[M]. 北京:清华大学出版社,2007
LI Gang. Application development of integration of Struts + Hibernate + Spring [M]. Beijing:Tsinghua University Press,2007

[3] 王福强. Spring 揭秘[M]. 北京:人民邮电出版社,2009
WANG Fuqiang. Unveil Spring[M]. Beijing: Posts & Telecom Press,2009

[4] 王霓虹,金兴. Ajax 技术及其 DWR 框架实现[J]. 自动化技术与应用,2007,26(12):92-94
WANG Nihong,JIN Xing. The Ajax techniques and DWR frame realization[J]. Techniques of Automation & Application,2007,26(12):92-94

[5] 许川佩,张民,张婧. 基于 Ajax 的 J2EE 安全应用框架[J]. 计算机工程,2010,36(4):110-111
XU Chuanpei,ZHANG Min,ZHANG Jing. Ajax-based J2EE security application framework [J]. Computer Engineering,2010,36(4):110-111

[6] 王国辉,王毅,尹相群. Java Web 开发技术方案宝典[M]. 北京:人民邮电出版社,2008
WANG Guohui,WANG Yi,YIN Xiangqun. Java Web development technology and solutions [M]. Beijing: Posts & Telecom Press,2008

[7] Johnson R. Spring Framework reference documentation[EB/OL].

[2010-08-28]. http://www.springframework.org/documentation
[8] 赵艳妮,王映辉,雷宇. 一种基于 AOP/IOC 的软件框架研究与实现[J]. 计算机工程与应用,2008,44(29):92-95
ZHAO Yanni, WANG Yinghui, LEI Yu. Research and implement of software framework based on IOC/AOP[J]. Computer Engineering and Applications, 2008, 44(29): 92-95
[9] 蒋纬,马光思. Spring 与其他框架整合及流程分析[J]. 计算机工程,2007,33(14):79-81
JIANG Wei, MA Guangsi. Integration and process analysis of Spring and other frameworks[J]. Computer Engineering, 2007, 33(14): 79-81

Research and application of Web framework based on SSH + DWR

CHEN Yao[1] LI Shan[2] ZHAO Yingnan[1]
1 School of Computer and Software, Nanjing University of Information Science & Technology, Nanjing 210044
2 College of Economic Management, Nanjing University of Aeronautics and Astronautics, Nanjing 210016

Abstract A Web development framework is helpful to improve the efficiency, maintainability and extensibility of web application. Integrating the layer design and some popular framework technologies, this article describes the design of a new Web framework based on SSH + DWR, in which Strusts decouples the View layer from the business layer, and Hibernate decouples the business layer from the persistence layer by the DAO mode, and Spring promotes a loose coupling between software components by IOC, and DWR improves the response of the user interface. Then an example is given to illustrate this new Web application, which fully reveals the strong extension, reusability, and sensitivity of client interface.

Key words SSH; DWR; Web framework