

MVC 在 J2EE 框架中的应用研究

隋 永, 周家纪

(成都理工大学 信息工程学院, 四川 成都 610059)

摘 要: 基于 B/S 的软件开发需要一种科学的软件开发模式, MVC 模式的设计思想为软件的健壮性、可维护性和可扩展性提供了有力的支持。文中将探讨 J2EE 框架与 MVC 模式相结合的开发方式, 并通过介绍基于 J2EE 与 MVC 模式所开发的企业管理系统来说明该模式的优点。

关键词: J2EE 框架; MVC 设计模式; 业务逻辑; 数据访问对象

中图分类号: TP311.5

文献标识码: A

文章编号: 1673-629X(2006)12-0119-03

Application Study of MVC in Frame of J2EE

SUI Yong, ZHOU Jia-ji

(School of Information Engineering, Chengdu University of Technology, Chengdu 610059, China)

Abstract: Software development based on B/S needs a scientific development model. The thinking of MVC makes a solid foundation for the software robust, maintenances, and extension. Discuss the combination of J2EE frame and MVC, and illustrate the advantage of this model by introducing the enterprise management system based on J2EE frame and MVC.

Key words: J2EE; MVC; BO; DAO

0 引言

随着 Web 技术的广泛应用, 基于 B/S 的软件开发得到快速的发展, SUN 公司所提出的 J2EE 框架成为 Web 开发的一支重要力量, 但基于 B/S 架构软件的健壮性、可维护性和可扩展性成为必须解决的问题。MVC 模式的提出使这些问题迎刃而解。SUN 公司在 J2EE 蓝图中建议在设计应用时使用 MVC (Model - View - Controller, 模型 - 视图 - 控制器) 设计模型。使用 MVC 的关键在于将逻辑分离为 3 个不同的单元: Model, View and Controller, 之所以分为这 3 个部分, 主要是因为应用数据结构和逻辑 (模型) 往往是应用的最稳定的部分, 而数据的表示 (视图) 的变化则可能相当频繁。如许多网站为跟上 Web 设计的最新形势经常改头换面, 但所使用的数据 (视图) 却没有变化。或者开发者可能希望用不同的语言来表示数据, 以及向内部和外部用户分别提供不同的数据子集等等。这些开发的具体需要决定了 MVC 模型的结构。

1 MVC 设计模式

MVC 是 Model - View - Controller 的简称, 最早是由 Xerox PARC 在 20 世纪 80 年代为编程语言 Smalltalk - 80

发明的一种软件设计模式。MVC 的基本思想是将应用数据和业务逻辑、数据的表示以及数据的交互相分离, 即分离为不同的实体, 分别为模型 (Model)、视图 (View) 和控制器 (Controller)^[1], MVC 模型如图 1 所示。

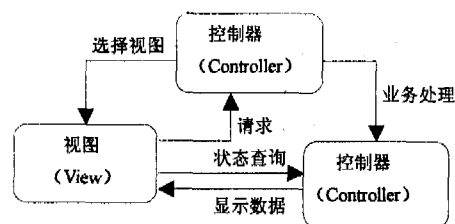


图 1 MVC 模型图

模型 (Model) 是应用程序的主体部分, 表示业务数据和业务逻辑, 可由其它组件进行访问。视图 (View) 是用户看到并为之交互的界面, 视图向用户显示相关的数据, 并能接受用户的输入数据, 但是它并不进行任何的业务处理。控制器 (Controller) 接受用户的输入并调用模型和视图去完成用户的需求。

MVC 模式的处理过程为首先控制器接收用户的请求, 并决定调用模型进行处理; 然后模型根据用户请求进行相应的业务逻辑处理, 并返回数据; 最后控制器调用相应的视图来格式化模型返回的数据, 并通过视图呈现给用户。由此可以看出 MVC 的优点为: 模型响应用户请求并返回响应数据; 视图负责格式化数据并把它们呈现给用户; 业务逻辑和表示层分离, 同一个模型可以被不同的视图重用, 所以大大提高了代码的可重用性。模型层是单独

收稿日期: 2006-03-07

作者简介: 隋 永 (1981-), 男, 山东诸城人, 硕士研究生, 研究方向为软件工程、数据库理论与技术; 周家纪, 教授, 硕士生导师, 研究方向为数据库理论与技术、图形图像处理。

设计的,它和视图以及控制器层保持相对独立,由此系统所使用的数据可在不同的数据库之间移植,从而保证系统的可扩展性。

2 MVC 模式在 J2EE 框架中的应用形式

SUN 公司针对 MVC 模式先后制定了两种规范^[2],称为 JSP Model1 和 JSP Model2。Model1 只在一定程度上实现了 MVC,视图、控制器和模型的分离并不完全。Model2 真正地实现了 MVC 模型。它建议由 JSP 技术实现视图的功能,用 Servlet 技术实现控制器的功能,用 JavaBean 或 EJB 技术实现模型层。但在实际的开发中,由于所开发系统的复杂程度不同,J2EE 框架中的技术在 MVC 模式中所承担的角色也不同。

主要应该分为以下 3 种形式。

2.1 使用 JSP 和 JavaBean

这种开发模式适合一些比较小的系统(如图 2 所示)。同时它也是一种很适于测试新思想和建立系统模型的方法。使用通用组件以及一些特定于应用的 bean 和动作,这往往是得到一个可见效果的最快途径。一旦有关思想得到证明,而且开发者对问题有了更深的理解,就能够对实际的最终应用体系架构做出决定。采用这种模式开发系统的缺点在于系统将很难维护和扩展。

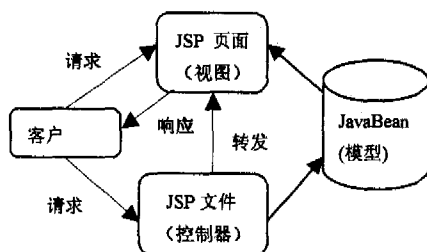


图 2 JSP 和 JavaBean 开发的模型图

2.2 使用 JSP,Servlet 和 JavaBean

这种开发模式对于一般的系统要求都可以满足,通常将 Servlet 作为应用的控制器,JSP 页面作为视图,此时 Servlet 相当于一个网关,它将请求分派至特定的处理组件,为客户请求选择适当的 JSP 页面进行响应。

采用这种模式的 MVC 角色的分配如图 3 所示,用户将请求发送至相当于控制器的 Servlet,Servlet 可以自行完成所请示的动作,也可以将每个动作委托至相应的各个 Bean 处理类,Bean 用于表示模型。根据处理的结果,控制器会选择适当的 JSP 页面来为用户生成响应(视图)。

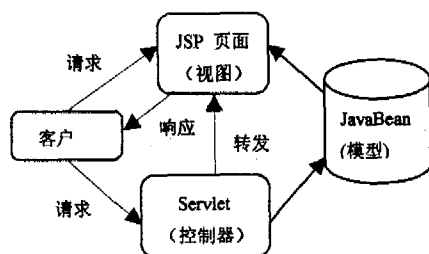


图 3 JSP,Servlet 和 JavaBean 开发的模型图

2.3 使用 Servlet,JSP 和 EJB

J2EE 框架中,EJB 是其最核心的技术^[3]。如今,基于 EJB 的应用通常也被看作是终极目标,EJB 所带来的的是事务管理和一个不依赖于客户类型的组件模型。通常当系统需求大量的数据库写访问操作的应用,而且要由多种不同类型的客户进行访问时可采用 EJB 开发。基于 EJB 的应用更强化了模型、视图和控制器的分离,从而可以得到易于扩展和维护的系统。EJB 定义了 3 种不同的企业 Bean,分别为:会话 Bean(Session Bean),主要负责系统业务逻辑功能的实现,又分为有状态会话 Bean 和无状态会话 Bean;实体 Bean(Entity Bean),表示存储在数据库中或某些其它持久性存储器中的数据,通常主要负责系统数据访问功能的实现,又分为 Bean 管理的持久化(BMP)和容器管理的持久化(CMP);消息驱动 Bean,它与会话 Bean 类似,只是需要 Bean 发送消息来调用它们。相应的角色分配如图 4 所示。

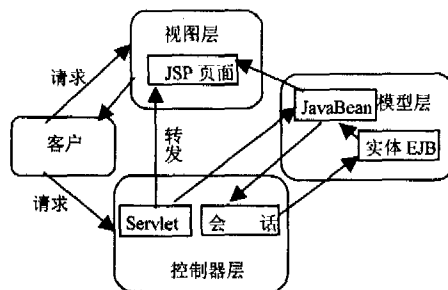


图 4 JSP,Servlet 和 EJB 开发的模型图

3 基于 J2EE 框架和 MVC 模式的公司管理系统

Web 技术的应用与现代化的管理方式推进了企业信息平台的诞生,目前越来越多的企业重视规划自己的资源,IT 界将其称为 ERP。企业信息化是指企业利用现代信息技术,通过信息资源的深入开发和广泛利用,不断提高生产、经营、管理、决策、服务的效率和水平,进而提高企业经济效益和企业竞争力的过程。具体到一个企业,企业信息化就是要实现企业生产过程的自动化、管理方式的网络化、决策支持的智能化和商务运营的电子化。企业信息化包涵 3 个方面的内容:一是产品设计和制造信息化;二是企业管理信息化;三是提供企业信息化的信息技术装备、产品和服务。

这个系统就是为企业管理信息化而开发的。它应用了基于 MVC 的 5 层体系结构,分别为 Web 层、控制器层、业务逻辑层、数据访问层、数据实体层,其中业务逻辑层、数据访问层和数据实体层为 MVC 模式中的模型部分(Model)。图 5 显示了系统的总体结构。

下面就各个部分说明它们的功能和实现的主要技术。

3.1 Web 层

这部分即 MVC 模型中的视图(View)。系统为员工登陆提供了统一 JSP 页面 login.jsp,系统提供了多种用户角色,员工可以根据自己的职位选择不同的角色登陆,据

此系统为不同的员工提供与其角色相对应的权限。本系统采用了 MD5 算法对用户密码进行加密,并将用户的登录信息放在 Session 对象中进行保存,简要代码如下:

```
users. userLogin (loginName, Common. Instance ( ). MD5Encrypt
(loginPws)); //用户验证
session. setAttribute("loginName", loginName);
session. setAttribute("login_ time", DateTime. Instance ( ). getCur-
rentTime(1)); //保存用户相关属性
```

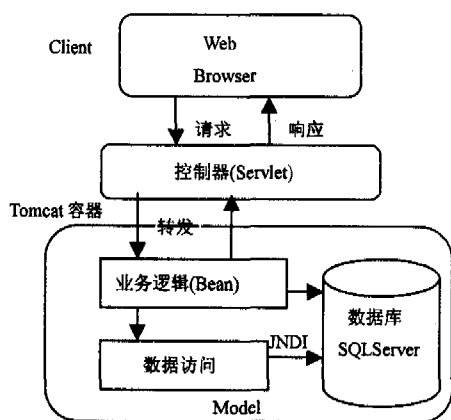


图 5 系统总体设计架构

用户身份得到验证后,即可进入系统的功能区,它由另外的 JSP 页面组成,在这里用户可以使用系统提供的各项功能。主要包括制定工作计划、发布公告、内部 BBS、业务资料的上传与下载以及提交工作建议等等。

3.2 控制层

这部分主要由 Servlet 负责,它相当于一个网关,通过在 Web.xml 文件中对 Servlet 文件进行配置,系统即可根据用户的请求信息自动地去调用相应的逻辑功能模块,并把处理后的信息发送到 JSP 页面呈现给用户。该系统采用了 Servlet 2.3 规范引入的两个组件类型:过滤器(filter)和监听器(listener)。过滤器可以截取发送至某 Servlet, JSP 页面或静态页面的请求,这样就可以很容易的将某一类的请求先提交给一个统一的 Servlet 进行处理。过滤器的配置形式如下:

```
<filter>
<filter-name>Control</filter-name> //过滤器名
<filter-class>
com.ora.jsp.servlets.Control
</filter-class> //过滤器的实现类
</filter>
<filter-mapping>
<filter-name>Control</filter-name>
<url-pattern>/login/* </url-pattern> //过滤器的匹
配形式
</filter-mapping>
```

监听器可以使系统应用对某些事件做出反应。我们在系统中为某些会话的激活(activation)和钝化(passivation)事件建立了监听器。监听器的配置形式如下:

```
<listener>
```

```
<listener-class>
```

```
Com.ora.jsp.servlets.ManagerListener //监听器的实现类
```

```
<listener-class>
```

```
</listener>
```

3.3 业务逻辑层(BO)

系统的所有业务逻辑都在这部分中完成^[4]。业务逻辑的主要功能为:

(1)接受 Servlet 所提交的请求,并根据编码的业务规则处理请求。

(2)使用数据访问层完成与数据库的交互任务。

这部分使用 JavaBean 技术实现,是整个系统中最为复杂的部分,需要大量的代码设计。但同时它也是最为重要的部分,业务逻辑 bean 的代码质量将直接影响到系统的性能和健壮性。该系统根据功能模块的不同将 bean 分配到不同的包中,这样在最大程度上提高了代码的重用性。

3.4 数据访问层(DAO)

数据访问层负责的任务主要是数据的持久化。这里所说的数据主要包括两个部分,一是一般的关系型数据,另外一种是对象数据库。在关系型数据库中存储对象数据存在阻抗不匹配(impedance mismatch)。这就要求使用数据访问对象(DAO)设计方式,该系统采用了 JNDI 配置数据库连接池,在 server.xml 文件中的部分配置代码如下:

```
<GlobalNamingResources>
<Environment name="simpleValue" type="java.lang.Integer"
value="30"/>
<Resource auth="Container" description="User database"
name="UserDatabase" type="org.apache.catalina.UserDatabase"
pathname="conf/tomcat-users.xml" factory="org.apache.
catalina.users.MemoryUserDatabaseFactory"/>
<Resource name="jndi/oa" type="javax.sql.DataSource"
password="" driverClassName="com.microsoft.jdbc.sqlserver.
SQLServerDriver" maxIdle="2" maxWait="5000" username="
sa" url="jdbc:microsoft:sqlserver://localhost:1433;Database-
Name=oa" maxActive="4"/>
</GlobalNamingResources>
```

系统同时使用了 Hibernate 来进行相应的对象-关系映射(ORM)。Hibernate 是一个面向 Java 环境的对象/关系数据库的映射工具,它提供了数据查询和获取数据的方法,这样可以大幅度减少开发时人工使用 SQL 和 JDBC 处理数据的时间。使用 Hibernate 最重要的是配置文件的设置,即 hibernate.properties 文件和类的映射文件 class-Name.hbm.xml。

3.5 数据实体层

这部分即系统所使用的数据库,本系统采用 SQLServer 作为数据库,其中包含了所有类的对应表,并根据在 Hibernate 的映射文件中所配置的信息,将表建立

(下转第 236 页)

表 4 实验结果

	Apriori 算法	FP-Growth 算法
生成的规则数	24	19
检测到的入侵事件	35	46
生成规则运行时间(s)	6.03	4.60

从上表可以看出 FP-Growth 算法在生成规则少于 Apriori 算法的情况下检测出了更多的入侵事件,有效地降低了规则的冗余度和系统的漏警率;同时生成规则所需要的时间也比 Apriori 算法要少。在处理实际入侵检测所产生的海量数据库和各种各样的入侵事件时,FP-Growth 算法的速度优势将更加明显地体现出来。从总体上来看,FP-Growth 算法的性能要优于 Apriori 算法,能够更好地为入侵检测系统服务。

4 结束语

文中研究了在入侵检测中应用 FP-Growth 算法。详细讨论了 FP-Growth 的实现方法,并对它进行了改进,实验证明改进的 FP-Growth 算法比传统的关联算法在入侵检测中的应用效果更好。但是我们注意到 FP-Growth 算法也没有检测出所有的入侵事件,因此进一步提高检测的精确度成了今后工作中应重点研究的内容。

(上接第 121 页)

了彼此间的关联关系。

下面是系统的整个工作流程,假设用系统管理员的角色登陆 login.jsp,系统将请求提交给 loginservlet,loginservlet 根据用户提交的信息调用业务逻辑层相关的 bean,并进行相应的帐户和密码验证,若通过则由 loginservlet 转发到 main.jsp,否则转发到 login.jsp。验证成功后管理员要添加人员,系统即调应 addperson bean,当管理员添加完信息提交后,系统将通过数据访问层在数据库的 person 表中添加相应的人员信息。

4 结 语

由上文可以看出,整个系统使用了 MVC 设计模式^[5],系统的主要优点是:

(1)在开发的过程中,只要定义好相应的接口规则,开发人员即可以专注于自己模块的开发,从而提高了系统的开发效率。

(2)Web 层与其它层相分离,可以根据当前的情况更新页面内容,增加系统需求的新功能,而其它的代码却无

(上接第 214 页)

子工业出版社,2003。

[2] TMS320C54X DSP CPU and Peripherals[M]. US:Texas Instruments. 1999.

[3] 赵红怡. DSP 技术与应用实例[M]. 北京:电子工业出版社,

参考文献:

- [1] HAN JIAWEI, PEI JIAN, YIN YIWEN, et al. Mining Frequent Patterns without Candidate Generation[C]//In: Proc Conf on the Management of data(SIGMOD'00, Dallas, TX). New York, NY, USA: ACM Press, 2000.
- [2] Agrawal R, Imielinski T, Swami A. Mining Association Rules between Sets of Items in Large Databases[C]//Proc Conf on Management of Data. New York, NY, USA: ACM Press, 1993:207-216.
- [3] 王新宇, 杜孝平, 谢昆青. FP-Growth 算法的实现方法研究[J]. 计算机工程与应用, 2004(9):174-176.
- [4] 吕 锋, 陈华胜. 关联算法的改进及其在审计数据挖掘中的应用[J]. 武汉理工大学学报:信息与管理工程版, 2004, 26(5):5-9.
- [5] 赵艳铎, 宋斌恒. 基于逆向 FP-树的频繁模式挖掘算法[J]. 计算机应用, 2005, 25(6):1385-1387.
- [6] 朱秋萍, 毛平平, 罗 俊. 基于关联规则的入侵检测系统[J]. 计算机工程与应用, 2004(26):160-162.
- [7] Bonchi F, Goethals B. FP-Bonsai: the ART of Growing and Pruning Small FP-trees: Proc 8th Pacific-asia Conference on Knowledge Discovery and Data Mining, PAKDD'04, Sydney, Australia[C]. Heidelberg, Germany: Springer-Verlag, 2004:155-160.

需太大的改变,从而保证了软件的可扩展性。

(3)业务逻辑层与其它层的分离,最大程度地提高了代码的重用性,只要对公共的组件进行优化,系统的整体性能即可得到明显的提高。

通过实践证明,该系统在 J2EE 框架中采用了 MVC 模式开发,极大地提高了系统的健壮性、可维护性和可扩展性。从而为许多基于 B/S 的系统提供了参考。

参考文献:

- [1] Bergsten H. JSP 设计[M]. 第 3 版. 北京:中国电力出版社, 2004.
- [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, 2005.
- [3] Roman E. 精通 EJB[M]. 第 2 版. 北京:电子工业出版社, 2002.
- [4] 易可可, 陈志刚. 基于 MVC 模式的 Web OA 系统设计与研究[J]. 计算机工程与应用, 2005(4):112-115.
- [5] 杨开英, 刘 树. MVC 设计模式在 J2EE 平台上的研究与实现[J]. 微机发展, 2004, 14(11):114-116.

2003.

[4] 汤华庚. TI54xxDSP 与 51 单片机的接口技术[J]. 单片机与嵌入式系统应用, 2004(1):27-30.

[5] 谢劲励. TMS320VC54X 接口技术应用研究[J]. 工业控制计算机, 2004, 17(3):17-19.