基于 SSH 框架的企业车辆管理信息系统①

张 蕾[1], 朱百祥[2], 唐永中[1]

[1](河西学院 信息技术中心, 张掖 734000)
[2](复旦大学 计算机科学与技术学院, 上海 200433)

摘 要: 企业车辆管理信息系统相对于传统的办公管理信息系统具有处理过程的不确定性和用户权限等独特的问题, 针对这些问题, 采用 SSH 框架设计了灵活的架构, 对用车申请和用车审批两个核心业务流程进行了设计与实现, 通过将策略模式和工厂模式结合起来, 实现了动态可热插拔的审批流程, 解决了不同用车申请需要调用不同审批流的问题. 对用户权限进行了设计以实现系统安全性.

关键词: 车辆管理; 信息系统; SSH; 设计模式; 框架

Vehicle Management Information System Based on SSH Framework

ZHANG Lei[1], ZHU Bai-Xiang[2], TANG Yong-Zhong[1]

[1](The Information Technology Center, Hexi University, Zhangye 734000, China)
[2](School of Computer Science, Fudan University, Shanghai 200433, China)

Abstract: The enterprise vehicle management information system has special characters compared to traditional management information system. For example, the process is uncertain according to different car apply request and the user privilege is different. SSH framework is used to design a flexible architecture. The design and implementation of car apply process and it's approve process are illustrated. To achieve the goal of flexible and pluggable approve process, design pattern of strategy and factory are combined. To achieve security, user privilege module is designed.

Key words: vehicle management; information system; SSH; design pattern; framewo

车辆管理信息系统[1,2]属于事务型办公管理信息系统, 常应用于企业基层. 用户在用车前必须填写相关表单提交申请, 经部门经理或公司领导的审批后, 交车辆管理部门进行分配司机、分配用车等出车安排, 在交车时对车辆的整体情况进行审查, 统计行车费用和车辆维修状况. 开发企业车辆管理信息系统可以提高企业车辆的有效管理, 合理分配用车以达到车辆的高效使用. 此外从车辆的正确保养和维修等角度出发, 可以提高车辆资料和驾驶员资料完整性, 保证车辆使用过程中各种信息的及时记录. 车辆管理费用的发生、车辆的检修工作、交车记录表的填写等操作通常统一在交车时进行统计, 这些统计数据有利于最终的管理决策. 企业车辆管理信息系统是办公管理信息系统中不可缺少的组成部分, 但又具有独特的设计难点, 比如: •处理过程的不确定性: 对于不同的用车申请可能会调用不同的审批流, 如市内用车只需要部门经理审核, 市外用车则需要公司领导的审核. •用户权限: 审于工作流的每个节点来说都需要有相关的用户进行操作, 其操作的权限要有明确的划分, 以保障数据的安全流转.

本文基于 SSH(Struts、Spring 和 Hibernate)框架设计并实现了车辆管理信息系统, 重点对用车申请和用车审批两个核心业务流程进行了设计与实现, 综合运用策略模式和工厂模式实现了动态审批流程, 并对权

① 基金项目:甘肃省教育厅高等学校研究生导师科研计划基金项目(0909-02)
收稿时间:2013-09-02;收到修改稿时间:2013-10-16

© 中国科学院软件研究所 http://www.c-s-a.org.cn

限管理模块进行了设计与实现.

1 相关技术背景

1.1 Struts

Struts 是基于 Model 2 之上的 MVC 框架, Model 2 将经典的 MVC 模型应用于 Web 应用, 同时根据 Web 应用中 HTTP 协议的无状态性对模型进行了变化. Struts 通过 ActionServlet 这一核心控制器拦截来自用户的请求, 并进而由用户开发的 Action 调用模型(由 ActionForm 和 JavaBean 组成)的业务逻辑方法处理请求, 最后将处理结果返回给 JSP 页面显示. 实现了模型和视图的分离, 使得层间保持松散耦合, 提高系统灵活性、复用性和可维护性.

1.2 Spring

Spring 是由 Rod Johnson 创建的开源框架, 它提供了轻量级的控制反转(IoC)和面向切面(AOP)的容器框架. 通过使用基本的 JavaBean 代替 EJB, 并提供了更多的企业应用功能, Spring 简化了企业应用开发, 并可以将简单的组件配置、组合成为复杂的应用.

1.3 Hibernate

Hibernate 是由 Redhat 公司负责维护的 Java 平台上一种全功能的、开放源代码的对象/关系映射(O/R Mapping)框架. 它对 JDBC 进行了非常轻量级的对象封装, 通过 HQL 查询语言, 或者使用 API 存储、更新和删除存储在数据库中的信息, 使得开发者可以使用对象编程思维来操纵数据库. 由于使用反射和运行时字节码生成, 它对于最终用户几乎是透明的.

1.4 SSH

SSH[3,4]是 Struts、Spring 和 Hibernate 三个技术框架的组合. 它将系统从架构上分为 Web 层、业务逻辑层和数据持久层. 在 Web 层使用 Struts 负责数据的获取和反馈; 在业务逻辑层使用 Spring 进行逻辑控制以及数据处理; 在数据持久层使用 Hibernate 框架完成应用程序与数据库之间的交互交换. 从而将三个框架的优势有机地结合在一起.

1.5 设计模式

设计模式[5]是经过分类编目的、可重复使用的代码设计经验的总结. GoF 设计模式包括 23 种设计模式[6-7], 分为三类: 第一类是创建型模式, 关注于对象的创建; 第二类是结构型模式, 关注于程序的结构化组织方式; 第三类是行为型模式, 关注于各个对象在运行时如何相互交互[8].

2 基于SSH的车辆管理系统

2.1 系统功能模块设计

系统的功能模块可以划分为系统管理、基础资料管理、出车管理、统计查询、我的工作台这几大分类, 各分类包含的模块和功能实现情况如图 1.

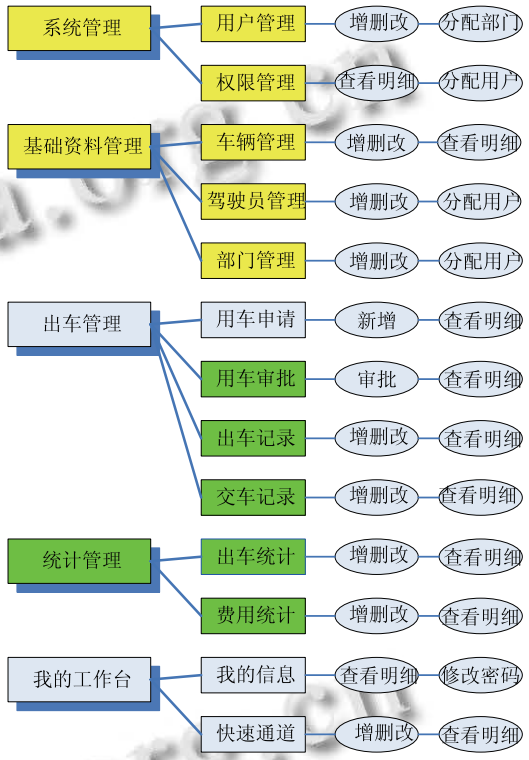

图 1 功能模块设计

2.2 系统架构

系统采用了基于 B/S 的分层软件架构, 浏览器为第一层, 作为系统的应用界面; 应用逻辑服务为第二层; 数据链接为第三层, 作为系统的数据存取服务. 该系统突破了传统分层方法, 在整体上采用了多层次的软件构架, 将该系统细化为客户展示层, 视图层, 控制层, 业务逻辑层, DAO 层, 实体层和资源层, 以便于系统复用和扩展, 如图 2 所示. 其中视图层, 控制层, 业务逻辑层, DAO 层和实体层是需要作为 Web 应用服务开发的部分.

系统采用 Struts 作为 MVC 框架, 借助 Struts 实现系统中的视图层和控制层, 利用 Hibernate 框架对持久层提供支持. 视图包括一组 JSP 文件, 利用 ActionForm Bean 来进行视图和控制器之间表单数据的传递, 通过

© 中国科学院软件研究所 http://www.c-s-a.org.cn

一组 Action 类的子类充当控制器，由 Spring 充当 Action 的代理，并通过控制反转来管理各个 Action. 采用 Hibernate 架构实现的 DAO 类来实现 Java 类与数据库之间的转换和访问，最后由 Spring 完成业务逻辑.

通过这样的架构，不仅实现了视图、控制器与模型的彻底分离，而且还实现了业务逻辑层与持久层的分离. 这样无论前端如何变化，模型层只需很少的改动，并且数据库的变化也不会对前端有所影响，大大提高了系统的可复用性. 而且由于不同层之间耦合度小，有利于团队成员并行工作，大大提高了开发效率.

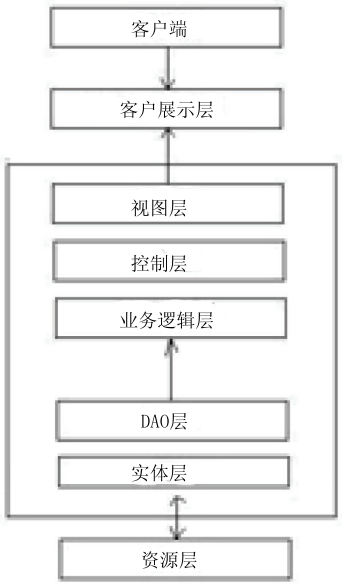

图 2 系统分层架构

2.3 数据库及持久化层设计

系统数据库设计了用户表、部门表、权限表、司机表、车辆表、用车申请表、用车审批表、出车表、交车表和费用表，表与表之间的关系图如图 3.

系统的持久化层采用 Hibernate 作为中间件，并使用了 DAO 设计模式实现对数据层的访问. DAO 模式是 J2EE 核心模式中的一种，其主要的行为就是在业务核心方法和具体数据源之间再增加一层，用这一层来连接业务方法和数据源，这样就实现了两者的解耦. 通过使用 DAO 模式，业务核心部分就不用关心数据层是如何实现对数据库的操作的，而只关心自己的业务操作，对数据库的操作全部交给了 DAO 代理，每个数据库表都对应着一个持久化对象，这样就给予了开发者使用 OO 思想设计和开发的便利，同时也屏蔽了具体的数据库和具体的数据表、字段，消除了对数据库操作的硬编码在重用性上的弊端.

因为要使用 Hibernate 来进行对象的持久化管理，因此要为每个模型创建一个对象/关系映射文件，每个文件都是以*.hbm.xml 文件命名.

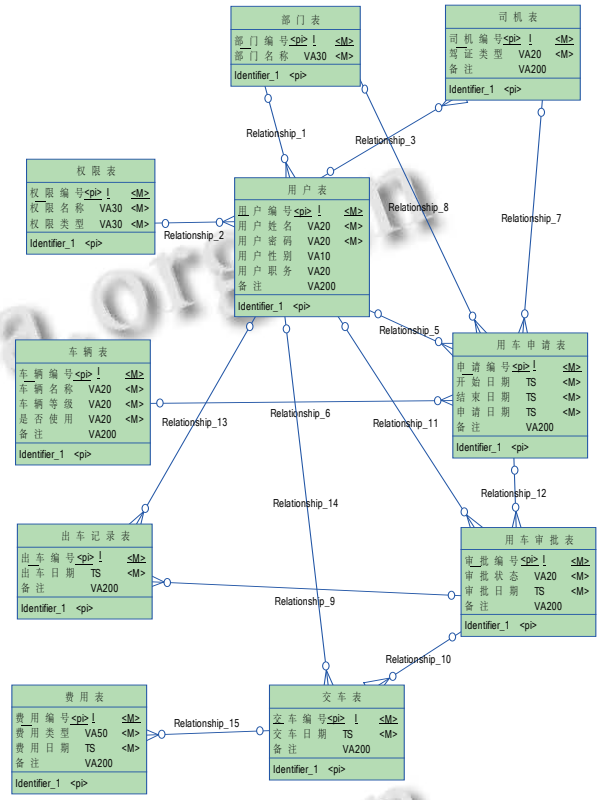

图 3 数据库设计

3 用车申请业务流程设计与实现

用车申请 carApplyAction 流程的主要功能是让用户填写用车申请表，并查看过去申请的审核结果.

流程如图 4:

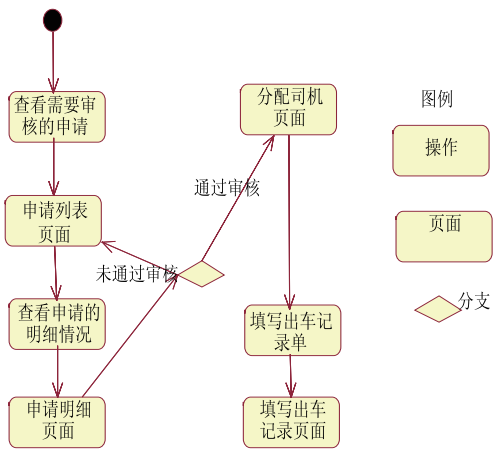

图 4 用车申请业务流程

© 中国科学院软件研究所 http://www.c-s-a.org.cn

该流程操作数据库中的 carapply 实体表和 applycheck 实体表, 申请时调用 addCarApply.jsp 界面填写申请表, 提交成功后系统会自动在两张表中均插入一条记录, 并将 applycheck 表中的审核状态默认设定为未审核状态(数据库中以英文字母 N 标示). 相关页面如表 1.

表 1　用车申请流程相关页面

Model	Carapply	汽车申请类
View	addCarApply.jsp	申请用车页面
	Declare.jsp	查询申请明细页面
	addCarApplyResponse.jsp	申请成功返回页面
Controller	DeclareAction.java	用车申请课控制器

在填写申请表页面, 利用迭代器 iterator 对未处于使用中的车辆进行迭代查询并将车名显示出来, 核心代码如下.

```
<td height="26">
<div align="center">
申请车辆:
<select name="carApply.cars.carsid" >
<ww:iteratorid="carInuses"
value="#session['carInuses']">
<option
value="<ww:propertyvalue="#carInuses.carsid"/>">
<ww:propertyvalue="#carInuses.carsname"/>
</option>
</ww:iterator>
</select>
</div>
</td>
```

为保证数据库数据的准确性和可读性, 本系统在数据库中将开始日期(startdate)、结束日期(enddate)和申请日期(applydate)都标示为 time stamp 类型, 故在输入的时候需要对用户输入的行为进行限制, 为此在页面上编写了一个 calendar()函数, 并在<input>标签内定义 onfocus="calendar()", 当用户鼠标定位到日期输入框时自动调用该函数让用户进行日期选择.

4　用车审批业务流程设计与实现

用车申请审批流程中的 applyCheckAction 的功能包括:

- 查看所有未审核的申请
- 对未审核的申请进行审核
- 对通过审核的申请进行分配司机操作
- 填写该申请相关的出车记录单

需要查询数据库中的 applycheck 实体表, 将所有未通过审核的记录显示在页面上; 通过审核后需要将表中的 checkstatus 字段修改为通过审核状态(数据库中以英文字母 Y 标示); 通过审核后需要调用 addCarApplyCheckSuccess.jsp 界面进行司机分配操作, 此处需要通过迭代查询出未出车的司机, 并将其姓名显示在下拉列表中; 分配成功后需要填写该申请单相关联的出车记录单, 记录实际的出车时间, 并在数据库中的 carout 实体表中插入相关记录.

针对不同用车申请需要调用不同的审批流的问题, 使用策略模式结合工厂模式进行了设计.

策略模式定义了一系列的算法, 并将每一个算法封装起来, 而且使它们可以相互替换. 将 applyCheckAction 针对不同的用车申请需要不同的审批流程的问题, 转化为根据不同的用车申请使用不同的审批策略/算法问题, 实现了可替换的审批流程. 如图 5, 审批流程抽象为 ApproveProcess_Strategy, 不同的审批流程如只需要部门经理审核的市内用车和需要公司领导额外审核的市外用车的审批流程分别封装在 UrbanApproveProcess 和 OutCityApproveProcess 中. 系统可根据管理者的配置使用不同的审批流程.

工厂模式可以将对象的创建和使用进行分离, 由一个工厂类根据传入的参数, 动态决定应该创建哪一个产品类. 系统将创建不同审批流程的职责分配给 ProcessFactory 类. 根据用车请求的不同自动创建不同的审批流程对象, 从而实现动态的、可热插拔的审批流程.

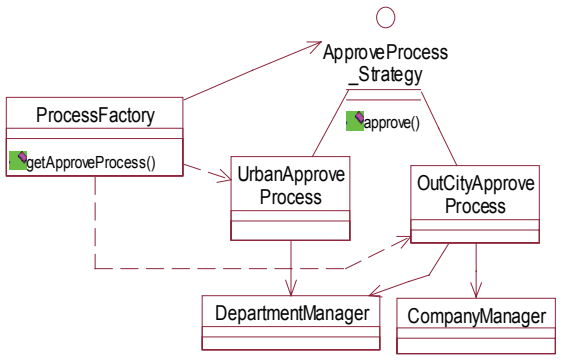

图 5　动态审批流程类图

© 中国科学院软件研究所 http://www.c-s-a.org.cn

5 权限管理模块的设计与实现

权限管理模块是办公管理信息系统安全性在应用层的体现, 它与数据库中的 privil 表相关, 通过权限的设置可以使不同的用户在登陆后对系统进行不同程度的使用. 如在本系统中系统权限用户可以使用全部功能模块; 审批权限的用户除了对系统管理类和基础资料管理类模块的访问权限受限外, 可以自由访问其他功能模块; 一般权限用户登陆后只拥有进行用车申请、申请结果的查看和对自己的密码进行修改的权限. 权限分配的情况直接对数据安全性产生影响, 提高系统的可靠程度.

本文的车辆管理信息系统中将权限主要区分为系统权限、审批权限和一般权限这三种固定的权限模式, 在权限管理模块中用户可以对当前权限的划分情况进行查看, 对系统的权限分配情况能有整体的直观把握, 但由于本系统采用了用户登陆后跳转进入不同主页的方式对用户进行权限隔离——即对于不同角色的用户, 在其成功登入后系统会自动判断并跳转到的相应的首页, 部分功能模块对于有些角色用户是不显示且/或部分链接无法使用的, 所以系统不允许在权限管理模块内对权限进行直接的新增、删除、修改等维护操作.

图 6 为权限管理的 UML 类图, 其中 PrivilAction 只包含了显示列表的功能, 如果要进行权限的维护则需要技术人员对页面进行相应的改动.

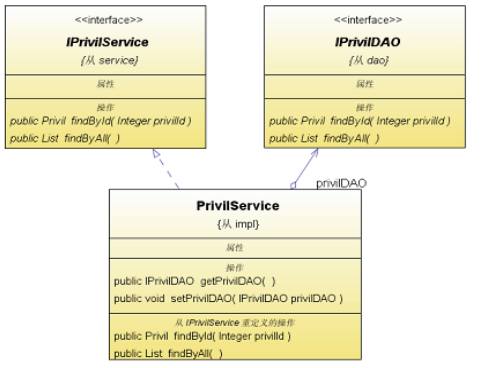

图 6 权限管理 UML 图

系统通过控制页面的判断, 为不同角色的用户加载不同控制侧边栏, 部分侧边栏分类对于非系统权限用户来说无法显示和/或链接无法使用. 加载不同侧边栏的页面代码如下:

```
<%@ page contentType="text/html;charset=UTF-8"%>
<%@taglibprefix="ww" uri="/webwork"%>
<ww:setname="privil" value="#session['privil']"/>
<ww:iftest="#privil.priviltype=='admin'">
<ww:includevalue="barAdmin.jsp"/>
</ww:if>
<ww:iftest="#privil.priviltype=='manager'">
<ww:includevalue="barManager.jsp"/>
</ww:if>
<ww:iftest="#privil.priviltype=='employee'">
<ww:includevalue="barEmployee.jsp"/>
</ww:if>
```

系统利用 SSH 框架对于页面加载的侧边栏进行控制, 如果需要对某个功能模块进行改动, 则直接对系统侧边栏的显示属性进行改动即可, 方便系统开发中的调试, 减轻代码强度.

6 结语

本文将 SSH 框架应用于企业车辆信息管理系统的开发, 降低了系统各层之间的耦合程度, 提高了系统的灵活性和复用性. 通过将不同的审批流程看作审批策略, 并将工厂模式与策略模式相结合, 实现了动态可热插拔的审批流程. 通过权限管理模块的设计, 保障了系统中数据的安全流转.

参考文献

1 钟岚,汪永超,毛明刚等.基于 B/S 的通用车辆管理信息系统研发.计算机工程与设计,2007,28(7):1695–1698
2 栾涛.GPS 消防车辆监控管理系统设计研究.科技通报,2012,28(4):168–170.
3 陈湘倩,狄文辉,孙冬等.基于 SSH 框架与 AJAX 技术的 Java Web 应用开发.计算机工程与设计,2009,30(10):2590–2592, 2596.
4 王宝龙,李子扬,李晓辉等.基于 SSH 框架和 DWR 技术的减灾卫星运行管理系统建设.计算机工程与设计,2010,31(23): 5096–5099.
5 陈建华,朱兰娟.基于设计模式的 Raid Cache 软件重构.微型电脑应用,2011,27(8):45–46.
6 Cederholm D. Bulletproof Web Design.北京:清华大学出版社.2006.
7 Rebecca M. Riordan. Head First Ajax.北京:中国电力出版社.2010.
8 彭宝琴,罗晓沛.基于 J2EE 轻量级框架组合的消费信贷系统的实现.计算机工程与设计,2008,29(3):647–649,674.

© 中国科学院软件研究所 http://www.c-s-a.org.cn