基于 Java EE 5 的房地产管理系统设计与实现

Design and Implementation of A Real Estate Management System based on Java EE 5

周明哲 王剑英 （中国科学技术大学 计算机科学技术系 安徽合肥 230027）

摘要：本文介绍了 Java EE 5 平台的主要特点和房地产交易、管理的一些具体流程，以及该系统设计和实现中的关键技术。本系统已应用到房产管理局的房地产信息管理系统中，运行稳定、高效。

关键词：房地产管理系统 Java EE 5 EJB 3.0 对象/关系映射

1 Java EE 5 技术介绍

Java EE 5（Java Platform，Enterprise Edition 5）是 J2EE 的升级版本，是 Sun 公司提出的一种基于 Java 技术的企业级应用模型。它提供了多层分布式的应用模型、组件复用、一致化的安全模型以及灵活的事务控制。在此基础上开发的系统具有较高的可用性、安全性、可扩展性和可移植性等优点。Java EE 5 技术内容庞杂，包括 JSP、Servlet、JDBC、JNDI、EJB 3.0、RMI、XML、JTS、JTA、JAF、Java Mail、JMS、Java IDL 等一系列基于 Java 的技术。一个典型的 Java EE 5 应用平台包含以下三层结构：

（1）表示层（Presentation Tier）：提供用户界面。

（2）业务逻辑层（Business Logic Tier）：进行逻辑判断、执行业务流程处理或提供公共服务。

（3）数据持久层（Data Persistence Tier）：实现到数据库的对象/关系映射（Object/Relational Mapping）；通过这个映射访问数据库，存储业务数据。

和 J2EE 相比，Java EE 5 主要具有如下新特性：

（1）Annotation[1]的使用。Annotation 中文意为标注，是 Java SDK 5.0[2]中引入的新机制之一，作用是通过在 Java 代码中加入元数据（Meta Data），影响编译器对代码的处理和程序运行时的行为。Annotation 可以用于：定义和使用 Web Service、开发 EJB 组件、映射 Java 类到 XML、映射 Java 类到数据库、指定外部依赖、指定部署信息包括安全属性等。有了 Annotation，XML 部署描述符不再是必须的，Java EE 5 应用程序的打包从而变得简单起来，如我们熟知的 WAR（Web Archive）应用中的部署描述符 WEB－INF/web.xml 将不再需要，部署信息通过 Annotation 进行描述，应用部署变得更加灵活。引入 Annotations 大大降低了企业级 Java 应用的开发成本，减少部署的步骤。

（2）EJB 3.0 的引入[3]。EJB（Enterprise JavaBeans）是一种企业应用技术，旨在建立一个企业应用开发框架。EJB 从第一个版本发布之日起就引起了极大的争议，因为 EJB 应用过于复杂。在 EJB 3.0 中，这一情况终于得到改善。EJB 3.0 不再需要 EJB home 接口，也不需要实现 javax.ejb.SessionBean 接口，一个简单的 POJO（Plain Old Java Object）对象就足以代表一个实体，并支持继承和多态。同时，困扰人们多年的 EJB 部署描述符变成可选的。EJB 的持久接口（Persistence API）变得更加简化、轻量级，EJB 服务器的查找也变得更加简便，JNDI（The Java Naming and Directory Interface）不再是必须的。EJB 3.0 还使用了拦截器（Interceptor），在业务方法被调用前进行拦截，因而更加容易实现灵活的 AOP（Aspect Oriented Programming）。

2 系统的需求

2.1 系统功能概述

（1）商品房备案。由于商品房是第一次进行交易，房产信息第一次进入系统，所以需要房产管理系统实现以下功能：登记发商提供的房产测绘资料，对预销售进行备案；在预销售转为销售时进行审核和备案。备案后才能进行下一步的房产权属交易。

（2）房产权属交易。这一部分是整个系统的核心。以房屋的测绘资料信息为依据，同时查询房产登

记的备案信息，判断当前将要交易的房产的状态（是否被查封，或者其它不允许交易的原因）。综合这些信息，按照预先设定好的流程，为用户办理产权交易等各种业务。

（3）房产档案管理。对房产交易中所经过的业务流程及结果进行整理和归档。这些档案包括商品房备案业务中得到的房产信息资料，房产权属交易业务中产生的交易合同细节等资料，司法查封中的查封信息资料。

（4）司法查封。如果接到法院等执法机关的司法协助通知，系统必须对房产所有权实施冻结，防止非法转移。这一部分专门登记和管理被冻结的房地产的情况，它为交易的合法性审核提供依据。分为查封、解封和续封，共三种操作。

（5）系统管理。由于整个房地产管理系统是一个多用户的系统，各个用户负责各种不同的流程，所以我们必须提供管理用户名、密码以及用户业务权限的功能。针对房地产交易业务流程多变，业务内容不断增加，证件、表单多样的特点，还要提供对业务大类及其流程进行灵活控制，业务表单、报表和证件的打印格式进行设置的功能。

2.2 房产交易流程概述

通过上述分析，我们可以知道，商品房备案和房产权属交易是整个系统中最重要部分。下面简单分析这两个业务的流程。

2.2.1 商品房备案

商品房预销售，简称预售，是指房地产开发企业将正在建设中的商品房预先出售给买方，并由买方支付定金或者房价款的行为。如果商品房已经建成，那么上述过程就叫做销售。如图1所示，开发商必须先在房产管理局登记，通过材料审核，现场查看，材料备案，并且得到预售（或销售）许可证以后，才能正式进行商品房预售（或销售）。

2.2.2 房产权属交易

房产权属交易是整个系统的核心。商品房交易、二手房交易、赠与、交换和继承等都归属在这一大类中。如图2所示，房产权属交易的流程包括查阅收取证件、登记受理、审核（多个审核步骤）、缮证、缴费、发放证件等：

（1）查阅、收取证件。在进行交易的时候，需要买卖双方提供各种证件和表单，例如交易申请表、房屋所有权证、买卖合同、房地产评估报告、土地使用权证等。收取的证件供下一步审核用。如果证件齐全，就接受交易申请，进入登记受理状态。

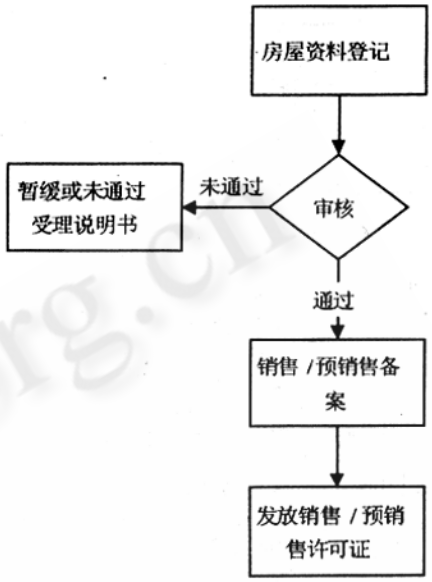

图 1　商品房备案流程图

（2）登记受理。这一步就是窗口的工作人员在接受交易申请以后，对交易双方提交的合同、产权证等材料进行整理，做好记录，并把申请材料和证件等一并交给审核人员。

（3）审核。在双方提交证件以后，需要对证件进行真伪验证，对卖方的资格进行审核等，这就是初审。当通过初审后，还要进行复审，也就是对初审意见、结果进行核实和确认。复审通过以后，进行最后的审批，确保交易无误的进行。如果审核没通过，就需要把流程步骤退回去，例如复审未通过，就退回到初审的状态重新审核。

（4）缮证。也就是打印、制造新的证件。在通过审核以后，旧的产权证、土地使用权证等会被收回。然后房产管理局会制造新的产权证和土地使用权证等。这些证件将在买卖双方完成缴纳税费这一步以后交给买方。

（5）税费收取。根据政策规定，房产交易中需要征收营业税、印花税、个人所得税等。除此以外，买方还要缴纳新证件的工本费。

（6）发放证件。在买卖双方完成上述步骤后，房产管理局将新的证件以及购房发票等发放给买方。交

易结束，由房产管理局将新的住房档案归档。

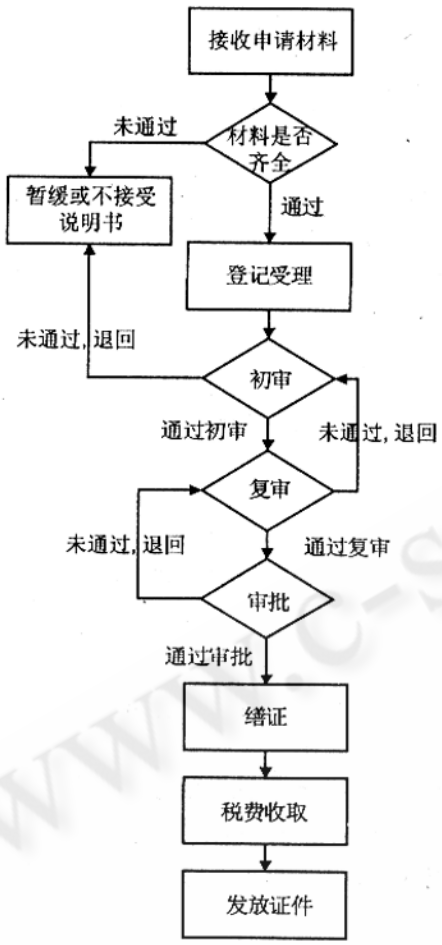

图 2 产权交易流程图

3 系统的设计与实现中的关键技术

3.1 系统模型结构设计

根据房地产管理的特点，该系统需要多用户、分散的进行各种房产管理业务的处理，而最终又需要把这些业务运作过程中产生的数据集中在一起，所以我们选择具有分布式特征的 B/S 结构。B/S(Browser/Server)结构，即浏览器和服务器结构。

用户工作界面是通过客户端的浏览器来显示，主要业务逻辑在服务器端实现，形成 B/S 结构。这样在客户端方面，就大大放宽了对计算机的要求。不管操作人员使用何种系统，如 Windows、Linux、甚至是通过 GPRS 上网的手机，只要有浏览器和 Internet 连接，就可以登录上服务器，处理各种业务。

在服务器方面，我们采用基于 Java EE 5 的三层结构，即表示层——业务逻辑层——数据持久层，见图 3。

表示层就是 Web 服务器(Tomcat)，采用 Struts 提供的 MVC(Model – View – Controller)框架[4]，其中的控制器(Controller)是一个 ActionServlet 类，它负责接收用户 HTTP(超文本传送协议)请求，通过业务逻辑层的会话外观(Session Facade)调用某一个特定的业务逻辑组件，并将得到的结果通过作为视图(View)的 JSP(Java Server Pages)页面返回给用户的浏览器。事实上，在整个系统里面业务逻辑层起到了 MVC 框架中模型(Model)的作用。

中间层是业务逻辑层。因为房地产的业务纷繁复杂，而且各种政策(例如税费、房贷利率、审核流程等)经常发生变化，所以我们有必要把整个业务操作分离出来放到独立的一层中，形成一个业务逻辑层，这样便于维护和升级。这一层负责接收表示层传递过来的各种业务的请求，例如商品房销售、租赁、档案管理等，然后执行具体的业务逻辑操作，必要的时候会对数据库进行访问，最后把操作结果返回给表示层。

最底层是数据持久层，包含了数据持久接口和数据库。数据持久接口构建在 JDBC(Java Database Connectivity)驱动的基础上，完成对象与关系数据库中实体的映射(Object/Relational Mapping)，实现数据实体的基本存取功能；数据库中记录了整个房地产管理系统中的所有数据，包括操作人员的用户名和密码、房屋测绘属性、房产交易历史、税费、房产抵押等数据。

3.2 系统实现中的关键技术

业务逻辑层和数据持久层中的数据持久接口位于同一个 EJB 3.0 容器，是整个系统的核心部分。这里的 EJB 3.0 容器采用 Oracle Containers for Java 10g(被 Oracle 公司简称为 OC4J)。

3.2.1 业务逻辑层

在业务逻辑层中，有四个业务逻辑模块：房产管理(包括房屋明细管理，商品房登记，司法查封管理)、房产权属交易(包括商品房、二手房交易和赠与、继承等业务)、系统流程设置(设置房产交易时所需要的各种流程步骤)和操作人员管理(管理操作人员的账号、登录密码和所属的部门等信息)。在四个业务逻辑模块的外面还有一个会话外观(Session Facade)[5]对各个模块进行封装，为表示层提供统一的调用接口，减少表示层和业务逻辑层之间的代码耦合程度。例如以下代

码片断：

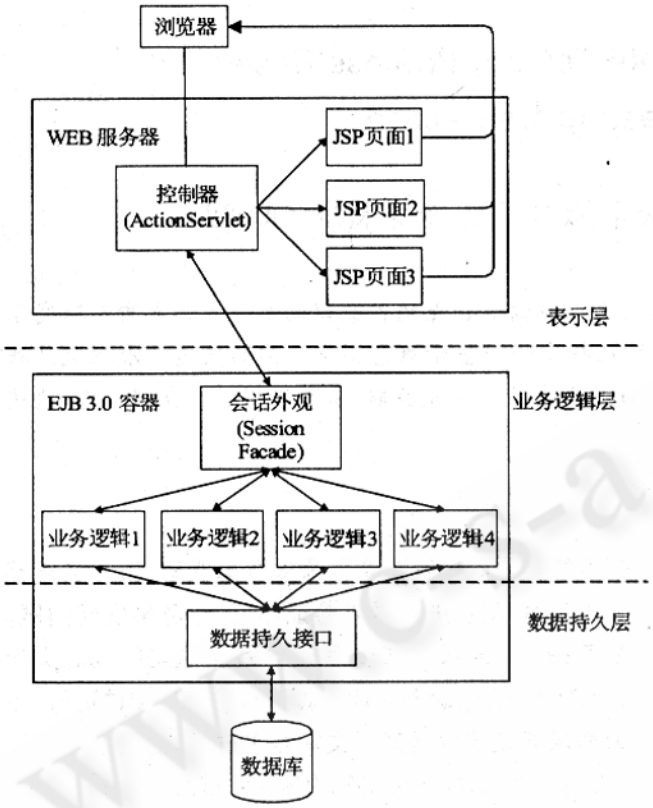

图 3　服务器端使用基于 Java EE 5 的三层结构

```
@Stateless
//表示该 Java 类是一个无状态的 Session Bean
public class UserSessionFacadeBean implement UserSessionRemote{
private BusinessLocal business;
private OperatorManagementLocal operatorM;
private SystemManagementLocal systemM;
private RealManagementLocal realM;
//以上四个成员是分别指向四个本地模块的引用
//……此处省略……
public boolean ProcessPaymentbyCash ( String customerid, double amount){//收取税费
business. ProcessPaymentbyCash ( customerid, amount );
//对 BusinessLocal 类中的方法进行封装
//为表示层提供统一调用接口
//避免表示层直接访问 BusinessBean
//以下代码省略……
}
}
```

在以上代码中，@ Stateless 带有@ 前缀。这就是一个 Annotation。Annotation 是 Java SDK 5.0 中引入的新特性。使用 Annotation 对类、方法和成员进行标记，使得 EJB 容器在运行时通过反射（Reflection）[6] 机制，获取 Annotation 中所标记的信息。

3.2.2　数据持久层

这一层由数据持久接口和数据库组成。数据库采用 Oracle Database 10g。在数据持久接口中，我们需要定义对象/关系映射（Object/Relational Mapping）。也就是将关系数据库中的一张表，映射成为一个 Java 对象。例如，为了将数据库中存储的操作员信息映射成为一个 Java 对象，我们需要编写一个类 RealUser，然后通过标记（Annotation）对这个类进行注解，从而使得这个类和数据库中的表 Real_User 相对应。以下是代码片断：

```
@Entity
//表示 RealUser 类的对象映射数据库中的表
@Table( name = " REAL_USER" )
//RealUser 类和数据库中名为 REAL_USER 的表对应
public class RealUser implements Serializable {
private String departId; //部门编号
private String password; //密码
private String userId; //用户 ID
private String userName;//用户姓名
@Id //@Id 注解的成员代表数据库中的主键
@Column( name = " USER_ID" , nullable = false)
//UserId 和数据库中的 USER_ID 列相对应
public String getUserId( ) {
return userId;
}//以下代码省略……
}
```

4　结束语

经实践证明，本系统运行稳定、高效。目前本系统已经成功的成为部分地区房产管理局的房产管理信息系统。

（下转第 80 页）

(上接第 75 页)

参考文献

1 Cay S. Horstmann, Gary Cornell. JAVA 2 核心技术卷 II：高级特性[M],机械工业出版社, 2006.

2 Sun Microsystems, Inc. JDK 5.0 Documentation [EB/OL]. http://java.sun.com/j2se/1.5.0/docs/, 2005.

3 Richard Monson – Haefel. Enterprise JavaBeans 3.0 [M]. O'Reilly, 2006.

4 Erich Gamma, Richard Helm, Ralph Johnson, et al. Design Patterns: Elements of Reusable Object – Oriented Software [M],机械工业出版社,2002.

5 Floyd Marinescu. EJB 设计模式[M], 机械工业出版社,2004.