摘 要

随着互联网日益深入社会生活，以网站作为对外的展示窗口，进行内外信息交流，已成为大众的迫切需要。本系统研究开发主要包括后台数据库的建立和维护以及前端应用程序的开发两个方面。对于前者要求建立起数据一致性和完整性强、数据安全性好的数据库。而对于后者则要求应用程序功能完备,易使用等特点。

基于房地产开发项目市场分析工作现状，不仅需要大量房地产楼盘（已开发完成或正在开）信息的积累，有时还需要有针对性地进行新楼盘的调查。目前在实际的工作中，一般采用 Mircosoft Excel 软件来存储已有房地产楼盘信息。为此，在进行具体项目的市场分析时，需要进行大量的手工操作才能得到有关分析结果。随着信息量的增大，工作效率越来越低，非常需要能够系统地存储并处理相关信息的信息管理系统。

本文主要就是研究如何将房地产信息采集、查询、统计以及电子地图定位完美的融为一体，并开发相应的系统。方便用户在查询房价信息的同时，可以精确的在电子地图中显示出相应楼盘的地理位置。信息采集使用统一的模板，统一的数据格式，提高了数据的准确性、可靠性。用户对采集来的基础数据进行统计分析，可以按区域、环线、区县来统计房价的平均售价以及环比指数、同比指数、首佳指数。这些统计结果为用户分析房价波动和宏观经济因素之间的关系、判断未来房价的走势，提供了很好的参考价值。

本文首先采用数据流图方法对客户的需求进行了结构化分析，并根据实际情况将系统分为 3 个部分：Web 服务、网站程序、客户端程序。Web 服务用来提供数据连接以及为第三方提供数据共享接口。网站程序提供信息上传的功能，用来收集房价、地价的基本信息以及各种宏观信息。客户端程序提供基础信息的查询、统计、地图定位、举报信息管理以及用户管理等功能。然后在结构化分析的基础上使用 ER 图方法进行了数据库的概念设计和逻辑设计。最后对系统各个功能模块进行详细设计，并完成系统开发。

关键字：房地产信息；电子地图；MapX；WebService；Asp.Net Ajax

Abstract

As the Internet becomes more in-depth social life, the Web site as the external display window, the internal and external information exchanges, has become the urgent needs of the public. Research and development of this system include the establishment and maintenance of front-end application development two aspects of the back-end database. For the former requirement to establish data consistency and integrity, data security database, For the latter requires the application fully functional, easy to use features.

Analysis of the status quo, based on the real estate development market not only requires a lot of real estate properties (developed or are open) the accumulation of information, and sometimes need to be targeted for new real estate survey. In the actual work, the general Mircosoft Excel software to store the existing real estate real estate information. To this end, during the market analysis of specific projects, the need for a large number of manual operations in order to get the results of the analysis. With the amount of information increases, the working efficiency is low, a great need for information to be able to systematically store and process information management system.

In this paper is to study how the real estate information collection, query, statistics, and the perfect positioning of the electronic map integration, and development of the corresponding system. User-friendly Check Rates information accurately in the digital map showing the geographical location of the corresponding real estate. The information collected using uniform template, a unified data format to improve the accuracy of the data reliability. The basis of data collected for statistical analysis, statistics by region, Link, districts and counties to the average selling price of the house prices as well as the chain index, an index, the first good index. These statistical results for the user to analyze the relationship between price fluctuations and macroeconomic factors to determine the future trend of prices provides a good reference value.

This article first structured analysis data flow diagram on customer needs,

according to the actual situation, the system is divided into three parts: Web services, web application, the client-side program. Web services used to provide data connectivity and data sharing interface is provided by third parties. The website provides information upload function used for the collection of housing prices, land basic information as well as a variety of macro information. Client program to provide basic information inquiries, statistics, maps, positioning, and report the information management and user management functions. And then in the ER diagram for the conceptual design and logical design of the database on the basis of structured analysis. Finally, the detailed design of each module of the system and complete system development.

Keywords: real estate information, electronic maps, MapX, WebService, Asp.Net Ajax

第1章 绪论

1.1 背景介绍

借助现代信息技术和管理理论，建立企业管理信息系统是当今社会的重要趋势。党和政府根据知识经济时代的特点，对国民经济建设提出了“用信息化带动工业化”的指导思想。对企业而言，全面开发和应用计算机管理信息系统就是近期不能回避的问题。在企业管理中，人力资源是企业最宝贵的资源，也是企业的“生命线”，因此人事管理是企业的计算机管理信息系统重要组成部分。而信息收集管理分析又是人力资源管理的重中之重。实行电子化的信息管理，可以让人力资源管理人员从繁重琐碎的案头工作解脱出来，去完成更重要的工作。

和其他的信息管理软件一样，房地产信息分析与服务软件是对各种房地产信息的收集、分析、统计，以便用户对相关信息做出进一步的评估、预测。但是房地产信息管理有其自身的特殊性，它需要将数量庞大的房地产信息以电子地图的形式直观的展现给用户，用户可以根据项目的具体地理位置和周边环境做出更准确的判断。

房地产的信息变化和国民经济的变化息息相关，包括银行、证券、抵押贷款等行业都会将房价的变化作为非常重要的参考依据，因此对房地产信息进行统计分析并形成可供各个相关行业参考的具体数据显得格外重要，而参考数据的可靠性、准确性需要建立在海量、真实数据基础之上。如何管理、分析这些重要数据并确保他们的安全就是本系统研究的主要目的。

软件的客户端功能可以概括如下：

1. 信息采集员分布在北京市各个区域，负责收集写字楼、住宅、商业、工业用地的详细情况，整理成 Excel 文件，然后通过登录本软件的网站，将 Excel 上传至服务器；

2. 数据维护人员使用该系统的网站，定期对提交上来的房地产信息进行校验，正确的、完整的数据将会被导入到服务器上的数据库中；

3. 数据分析人员通过本软件的 C/S 客户端程序，对数据库服务器上的各种数

据进行查询、分析、统计，得到各种统计图表和文档；

4.地图维护人员通过本软件的 C/S 客户端程序，将房地产信息添加到地图中。其他用户可以使用更新地图的功能，使本地的电子地图和服务器上的电子地图信息保持同步。

a) 和其他 GIS 系统的比较

百度地图、谷歌地图等系统可以提供准确的、海量的地图信息，但是要想在他们之上增加自定义的图层信息（商业、住宅、工业等）实现起来非常困难。另外考虑到房地产信息的特殊性——在建楼盘不会出现在百度地图、谷歌地图中，而房地产信息分析需要从地产项目立项就开始追踪分析，这就导致了本系统无法采用百度地图、谷歌地图等类似平台。

b) 地图放在服务器端还是客户端

地图的信息容量非常大，以北京市区为例，约 50 个 layers（图层）大约 700M，如果将这些地图信息放到服务器端，那么需要采取复杂的分割算法对地图进行处理，这对程序算法和服务器的要求都非常高。如果将地图直接打包安装到客户端那么可以极大的减少服务器的压力和网络流量并提高客户端的响应速度，只需要在地图自定义图层（商业、住宅、工业等）更新时，由客户端自动同步服务器端相对应的图层即可，另外考虑到本软件的使用者都是固定的付费用户，因此最终决定采用客户端这种实现方式。

软件的服务端功能可以概括如下：

1. 在统计分析大量真实数据基础之后，形成各种统计报表，报表的真实、直观可以作为银行等行业的参考数据，本系统通过 web service 接口可以将这些数据以不同的形式提供给银行，并通过硬件加密狗的授权方式确保服务端的安全；

2.任何遵守本软件接口的软件都可以向本平台中提供各个楼盘的真实数据（由服务端统一付费）。这样可以确保数据的及时性、准确性。

1.2 可行性分析

1.开发技术的可行性

该软件主要分为 3 大部分：

(1)网站：提供 Excel 数据上传、校验、导入、更新、删除；Word 文档的共

享、信息发布；

(2)C/S 客户端：将电子地图直接部署在客户端，用户可以更新地图，添加房地产信息，查询、统计数据库服务器的各种房地产信息；

(3)Web 服务：用来连接 C/S 客户端和数据库服务器；并将接口提供给第三方公司开发的系统，实现部分数据的共享；

通过上述3个部分，可以较好实现用户的全部需求，还能实现和另外一家公司的软件实现数据共享。综上所诉，从技术角度考虑，实现该房地产信息分析与服务软件是可行的。

2.经济可行性

硬件需求：保障本软件正常运行只需要配置一台中等性能的服务器，客户端可以利用公司现有的普通PC机即可。

软件需求：服务器安装 windows server2003 、SQL Server 2000 、.net framework2.0 、IIS ；客户端安装 windows XP 或者 windows vista 、.net framework2.0 ；

1.3 关键技术

1.3.1 ADO.NET 简介

每个MIS系统都要选择一种合适的数据访问方式。该房地产软件采用Visual Studio2010开发，选择ADO.NET进行数据访问。ADO.NET是微软最新的数据访问技术。

ADO.NET的设计目标：

1.简单的访问关系和非关系数据；

2.统一XML和关系数据访问，支持Internet上的多层应用程序；

3.与上一代技术（ADO）相比，它支持更多的数据源；

ADO.NET类和对象的概述：

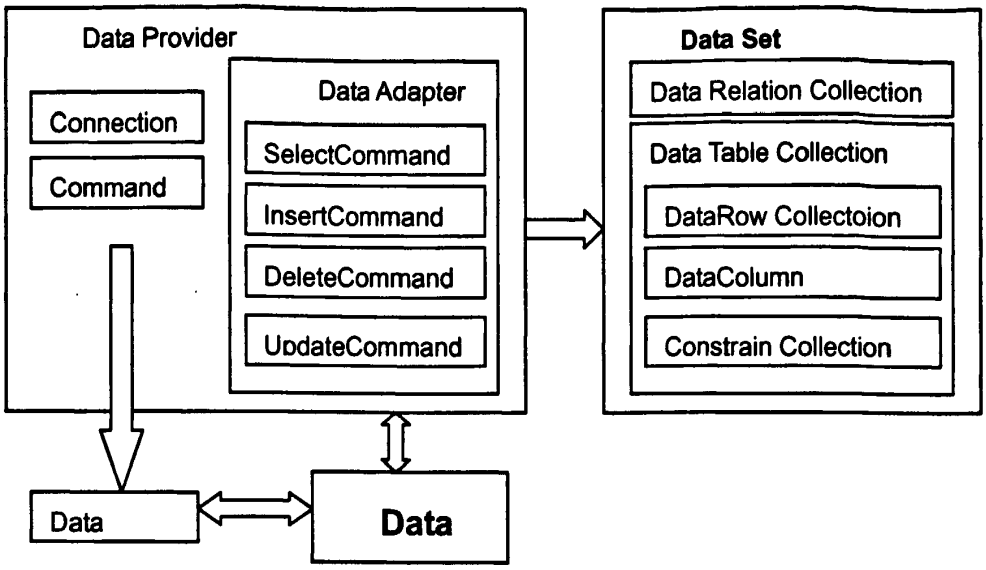

图 1-1 ADO.NET 的架构图

ADO.NET 的对象模型是很丰富的，然而它的核心只包括一些比较简单的类集。关键类是 DataSet，它用于表示一个数据库的富子集，它缓存在会话状态和内存中，且不需要不间断的数据库连接。DataSet 可以表示一系列表，并可以使用元数据来表示数据表之间的关系和约束。可以这么说，DataSet 就是本地内存中一个小的数据库。

ADO.NET 有丰富的数据提供程序，支持 OLE DB ，SQL Server，ODBC 连接。尤其是 ADO.NET 可以很容易的和 XML 进行互操作，这使得在更大的范围实现数据共享。

1.3.2 Web Service

Web 服务是松耦合的，它与服务器和客户端所使用的操作系统、编程语言都无关。与从前的分布式技术（如 DCOM、CORBA）不同，Web 服务不要求客户端和服务器端使用相同的语言，也不要求他们使用相同的操作系统。

Web 服务的这些特性不仅可以满足该房地产管理软件对数据访问的要求，也可以将 Web 服务的接口提供给另一家公司，实现数据的共享。

Web 服务架构包括以下特性：

1.Web 服务的服务器和客户端程序都能连接到互联网，客户端程序通过 Web 服务的本地代理实现远程过程调用（Remote Procedure Call，RPC）；

2.用于通信的数据格式遵循SOAP标准，SOAP消息是自我描述的、基于文本的XML文档组成；

3.客户端和服务器端的系统是松耦合的。只要Web服务和使用Web服务的应用程序都能够发送和接收遵循适当协议标准的消息就可以；

Web服务的缺点：

对于同样的数据，XML文档通常要比二进制的文件大得多。在Internet上传输大量的数据，如果带宽不够，那么会有明显的传输延时。但是如果在局域网内部，这种延时基本可以忽略。当然我们还可以采用数据压缩技术，Web服务在发送前将信息压缩，在客户端接收时再解压以达到减少数据传输量的目的。

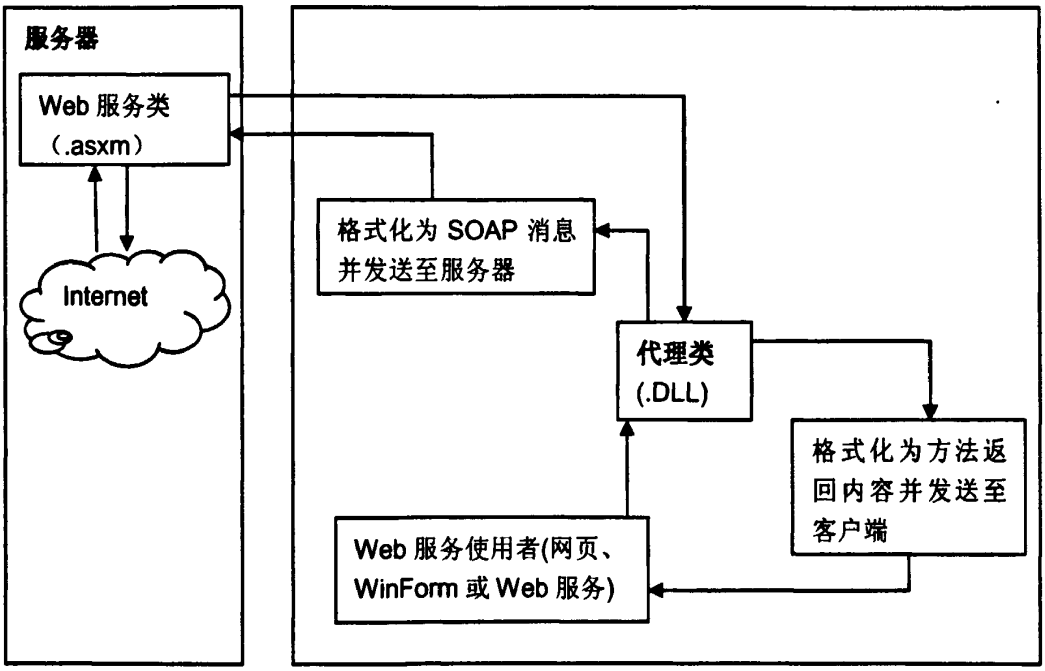

图1-2 Web服务运行原理图

第2章 需求分析

2.1 功能需求

XX 公司房地产信息管理是针对各种房地产数据的管理，包含上传、验证、查询、统计分析、地图编辑等功能。基础数据主要分为五种类型：监测点房价信息，市调信息，宏观数据信息、协议出让信息、招拍挂信息。其中，监测点信息需要实现地图编辑、显示、统计、分析功能；市调信息在房价监测系统中作为独立数据，实现简单的查询功能，并提供接口给第三方软件实现数据的共享；宏观数据作为独立数据，实现查询功能。协议出让信息、招拍挂信息实现地图编辑、查询、统计；

信息采集人员负责基本数据的采集，每月 10-15 日由每个监测组组长将本组的监测点信息以 Excel 文档的形式上传到本系统中，上传后由系统管理员校验确认，系统管理员将符合要求的 Excel 调查表格导入后台的 SQL Server2000 数据库中。每个季度每个监测区域形成一份分析报告（Word 文档），每季度第一个月 10-15 日由每个监测组组长将上个季度自己监测区域的分析报告上传到本系统中，由系统管理员校验确认后正式在系统中显示。

管理员负责电子地图的更新，将每个监测点（即楼盘）按照具体地理位置标注到电子地图上，用不同颜色的圆点表示，如：住宅-红色、写字楼-蓝色、商业-黄色、工业-绿色；对各种上传数据（Word 分析文档、Excel 模板数据）进行验证，将符合要求的 Excel 数据导入后台数据库，负责基本数据的更新、删除；对一般用户举报的信息进行核实、修订；管理员还负责用户及其权限的维护；

一般用户使用本系统可以查看各种基本类型的数据。当鼠标指向某一项目（楼盘、地块）时，可以看到这个项目的基本情况，如：项目名称、地理位置、物业类型、所属区域、销售价格和租金（显示最近一个月的数据）；当鼠标框选某个区域时，以列表形式显示所有被选中项目的基本信息，通过基本信息中的项目编号可以进一步查看某个项目详细信息，还可以查看某个项目的所有历史信息；当项目的售价、租金明显有误差的时候，可以进行举报；可以登录网站查看、下载各季度分析文档；

高级用户（经理）拥有一般用户所有的权利，还可以对各种基本信息进行统

计分析、生成各种报表。按区县统计均价、按物业类型统计均价、按监测区域统计均价、按调查日期统计均价并生成折线图、柱状图、饼图；计算房价的同比、环比、首佳指数，按上述数据查询历史房价指数走势并且以折线图的形式展现出来。

相关计算公式：

1.区域均价 = Σ(每个楼盘的销售价格×该楼盘建筑面积)÷所有楼盘总建筑面积

2.平均租金(单位：元/天.平方米)= Σ(每个楼盘的租金×该楼盘建筑面积×365)÷(所有楼盘总建筑面积×365)

3.环比 =(本月销售均价 － 上月销售均价)÷本月销售均价

4.同比 =(今年本月销售均价 － 去年本月销售均价)÷今年本月销售均价

5.首佳指数 =(本月平均销售均价 ÷ 2008年1月的销售均价)×100

2.2 模块划分

本软件分为三个子系统：网站程序，客户端程序，Web服务；

1.网站程序包含如下模块：

(1) Word、Excel文档的上传模块

将本地文档上传至服务器。

(2) Excel文档的数据校验、导入模块

对上传到服务器的数据进行校验，将符合要求的数据导入后台SQL Server2000数据库相对应的表中。并且确保导入操作的事务性。本系统所有基本数据的来源都依靠该模块实现。

(3) Word文档下载模块

提供用户下载各种分析文档。

(4) 数据的维护模块

管理员可以在线对已经导入数据库中的基本数据进行更新、删除。

2.客户端程序包含如下模块：

(1) 对房价基本信息的查询模块

房价的基本信息包含两种类型：监测点房价信息、市调信息。检测点房价信息按物业类型又分为住宅、写字楼、商业、工业4个类型。该模块实现对这些数

据的查询。

（2）对房价的统计模块

统计结果主要包括最新房价指数、历史房价指数、房价指数走势图。

（3）房价信息举报模块

对房价有明显误差的信息进行举报，交给系统管理员核实。

（4）对宏观信息的查询模块

宏观信息主要包括4个类型：主要宏观数据、房地产相关数据、商品房预售成交数据、商品住宅预售成交数据。其中房地产相关数据又包括开发完成投资情况、施工情况、竣工情况、新开工情况、空置情况、现房销售情况、期房准售情况。该模块主要实现对这些数据的查询。

（5）对地价信息的查询模块

地价主要包括两种类型的数据：协议出让信息、招拍挂信息。该模块主要实现对这些信息的查询。

（6）对地图的维护模块

实现电子地图的编辑（增加点、删除点）。对房价信息按不同的物业类型（住宅、写字楼、商业、工业）分层（Layer）标注项目信息，对地价信息按协议出让、招拍挂分层标注项目信息。实现地图的放大、缩小、平移、点选、框选、圈选、不规则选择等基本操作，提供鹰眼实现地图的快速定位。当某个终端用户更新地图后，其他用户可以更新本地的电子地图。

3.Web服务

Web服务部署在服务器端，它不但提供了客户端的数据查询要求，还将接口提供给第三方软件，实现市调信息的共享。

2.3 软件架构图

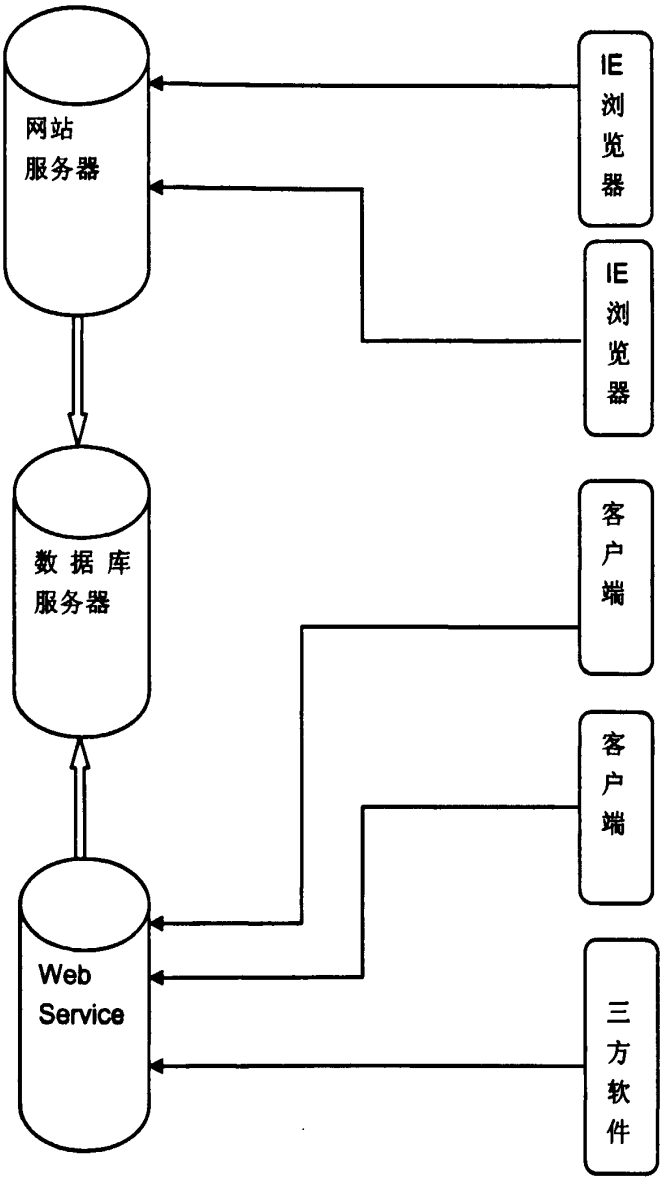

图 2-1 软件架构图

2.4 数据流图

数据流图（Data Flow Diagram）：简称 DFD，它从数据传递和加工角度，以图形方式来表达系统的逻辑功能、数据在系统内部的逻辑流向和逻辑变换过程，是结构化系统分析方法的主要表达工具及用于表示软件模型的一种图示方法。根据层级数据流图分为顶层数据流图、中层数据流图和底层数据流图。除顶层数据流图外，其他数据流图从零开始编号。

顶层数据流图只含有一个加工表示整个系统；输出数据流和输入数据流为系

统的输入数据和输出数据，表明系统的范围，以及与外部环境的数据交换关系。中层数据流图是对父层数据流图中某个加工进行细化，而它的某个加工也可以再次细化，形成子图；中间层次的多少，一般视系统的复杂程度而定。 底层数据流图是指其加工不能再分解的数据流图，其加工成为“原子加工”。

为便于管理和阅读，要对每个层次上的图及其加工进行编号。层次编号自上而下分别为顶层图（系统图）、0 层图、1 层图、等等。各层图的关系为父子关系，下层图为子图，上层图为父图。

一个数据流图确定了系统的转化过程、系统所操纵的数据或物质的收集（存储），还有过程、存储、外部世界之间的数据流或物质流。数据流把层次分解方法运用到系统分析上，这种方法很适用于事物处理系统和其他功能密集型的应用程序。

经过分析，得出的数据流如下图所示。

2.4.1 顶层数据流图

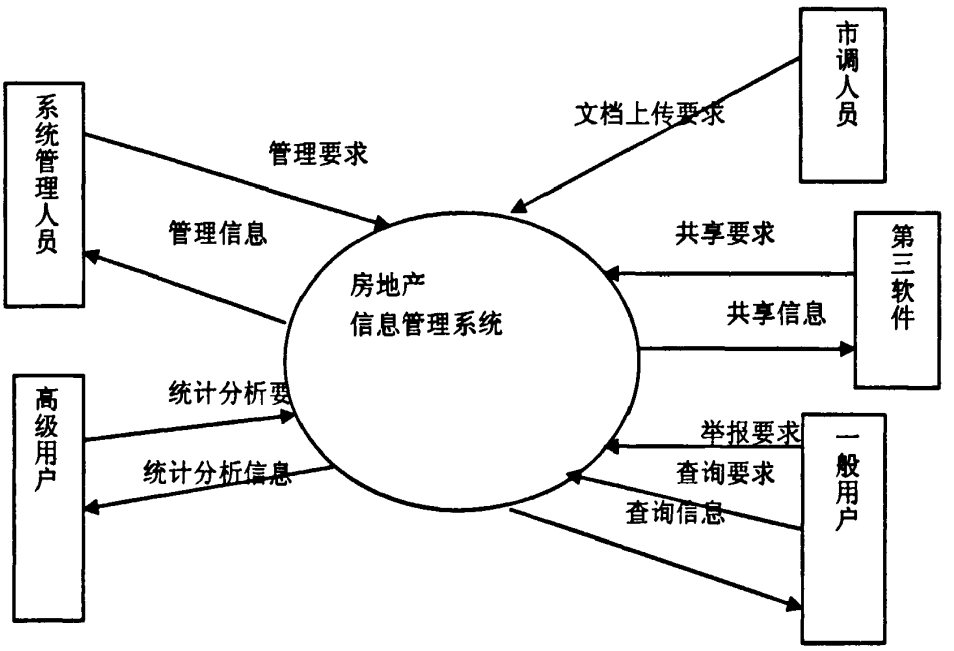

图 2-2 顶层数据流图

图 2-2 描述了系统管理人员、高级用户、一般用户、市调人员、第三方软件

五种不同角色的人员对软件提出的使用要求。顶层数据流图是对系统概括的抽象。它标明了软件功能的边界，为进一步细化软件功能提供了基础。

2.4.2 0 层数据流图

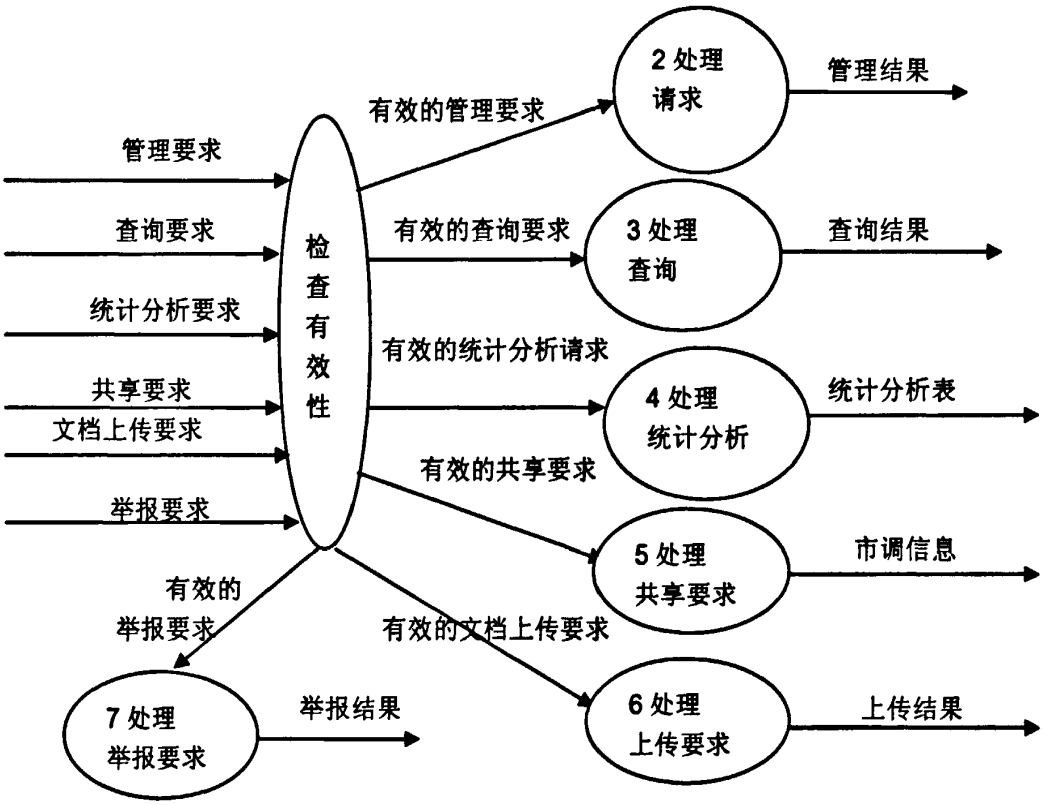

图 2-3 0 层数据流图

2.4.3 1 层数据流图

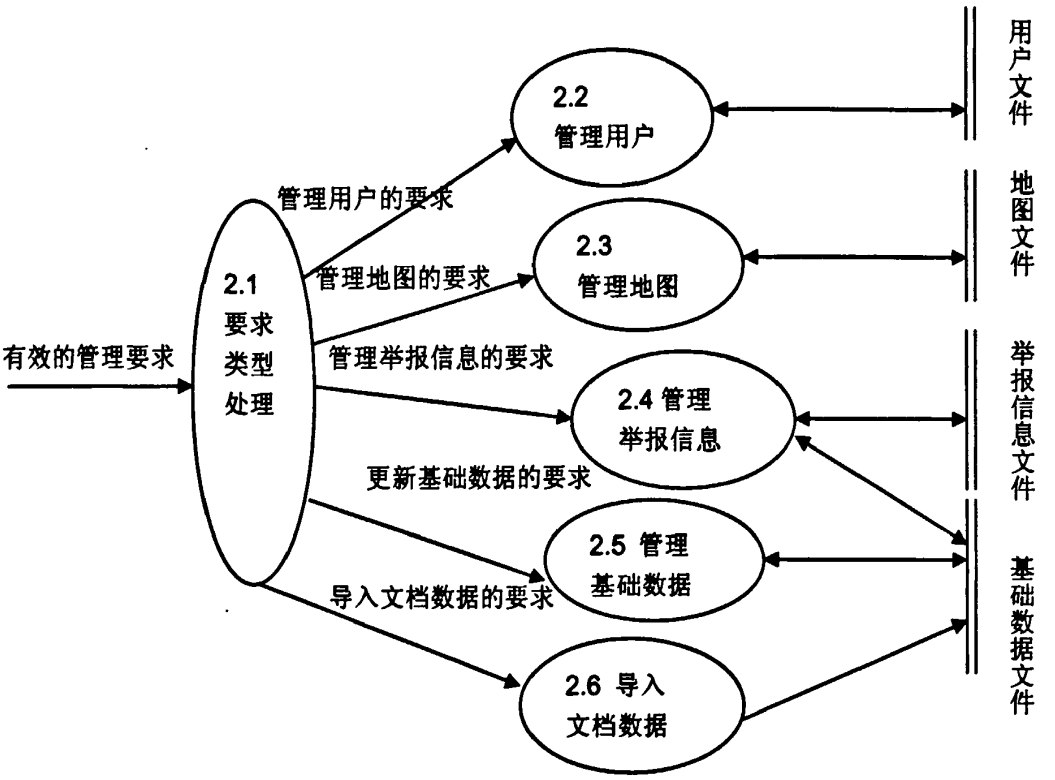

图 2-4 一层数据流图

图 2-4 描述了系统管理员对软件提出的各种功能要求。包括用户管理、地图管理、举报信息的管理、基础数据的管理以及将 Excel 文档导入数据库的要求。

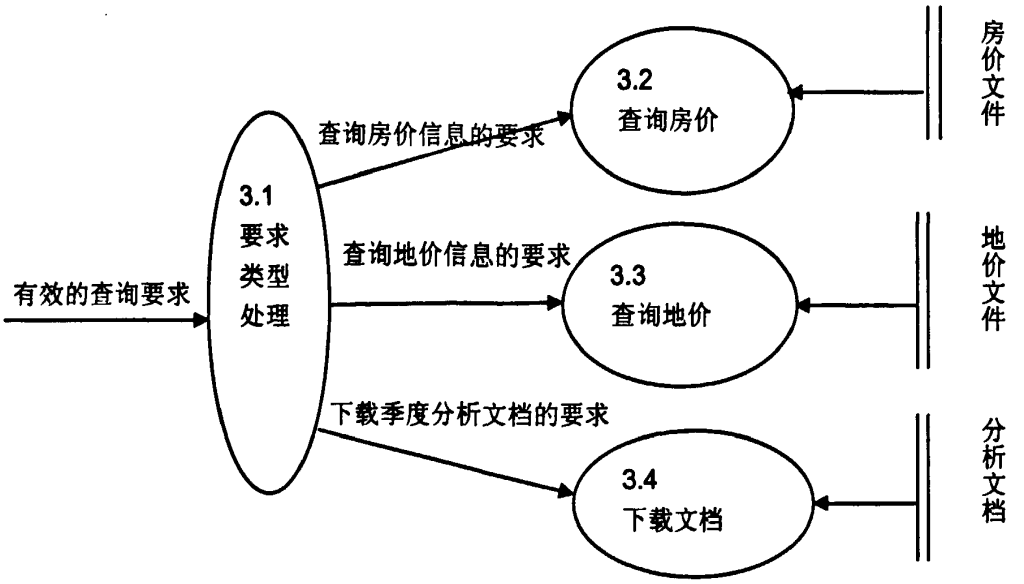

图 2-5 一层数据流图（查询要求）

图 2-5 描述了一般用户对系统提出的查询要求。

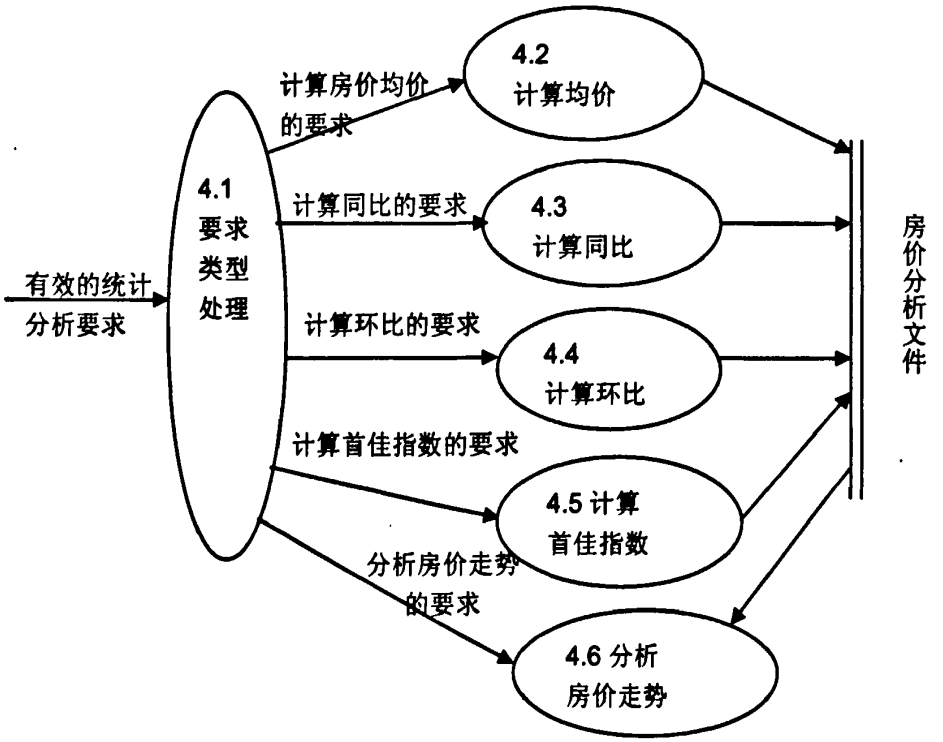

图 2-6 一层数据流图（分析要求）

图 2-6 描述了高级用户对软件提出的各种需求。根据各种计算公式按时间计算均价、同比、环比、首佳指数并将统计结果以曲线图的形式展现给用户。

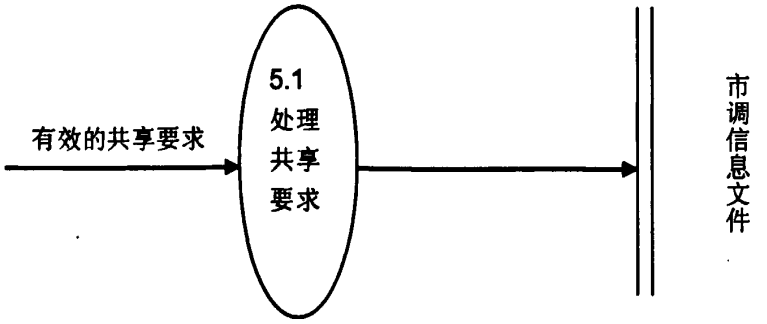

图 2-7 一层数据流图（共享要求）

图 2-7 描述了第三方软件提出的共享数据的要求。系统通过 Web Service 提供给第三方，实现数据的共享要求。第三方用户只需要在本地引用 Web Service 的本地代理就可以访问位于数据库服务器中的市调信息文件。

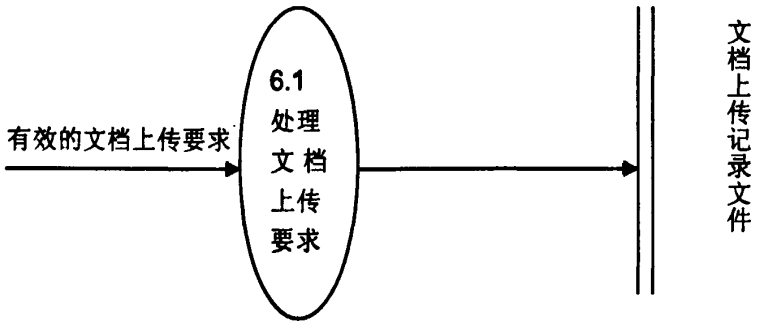

图 2-8 一层数据流图（文档上传要求）

图 2-8 描述了信息采集人员对软件提出的需求。信息采集人员通过登录网站将各种 Excel 表格上传至服务器，等待系统管理员的处理。

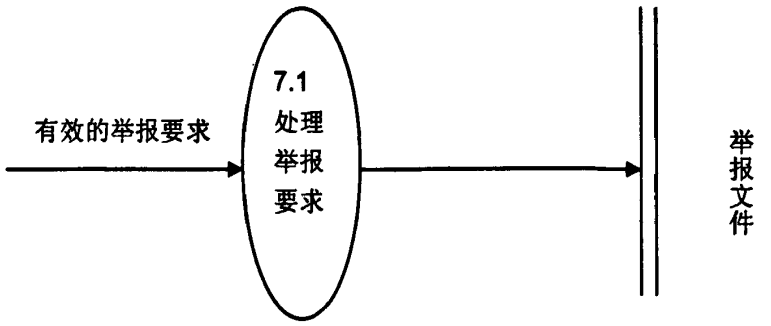

图 2-9 一层数据流图（举报要求）

图 2-9 描述了一般用户在使用软件过程中，发现房价基本信息的敏感字段（如售价、租金）和实际情况有差异时，需要将基本信息举报给系统管理员，并请求系统管理员进行处理的要求。

2.4.4 2 层数据流图

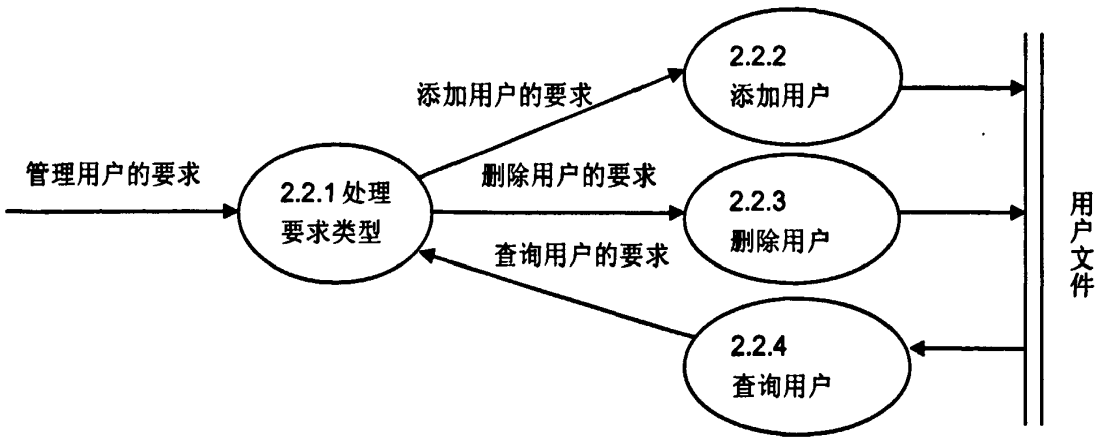

图 2-10 管理用户的二层数据流图

图 2-10 描述了超级管理员 Admin 管理软件使用人员的要求。除添加、删除外，管理员还可以查询用户使用软件的总次数和最后一次的登录日期。在添加用户的时候管理员还可以指定用户的角色。

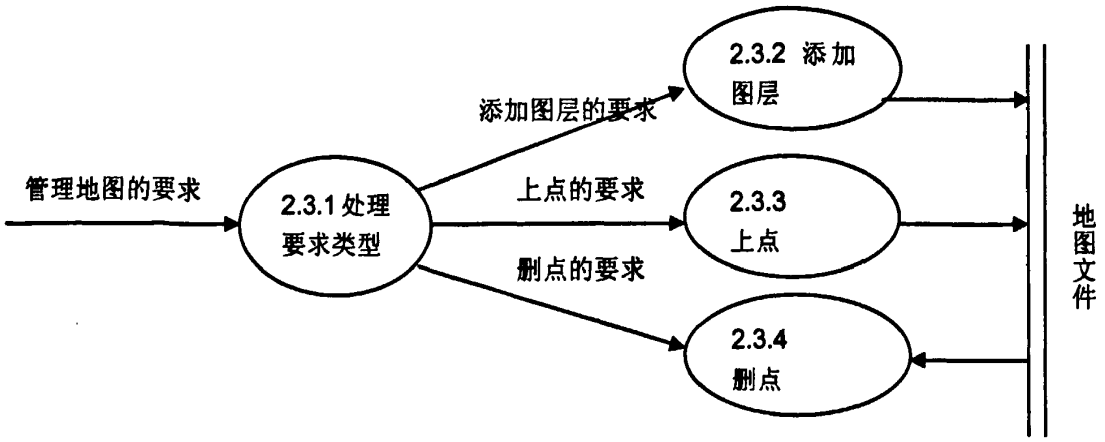

图 2-11 管理地图的二层数据流图

图 2-11 描述了管理员管理软件中电子地图的要求。管理员使用软件对地图进行更新，增加新的房价、地价信息。为了区分不同物业类型房价信息和不同种类的地价信息，将他们分别保存在不同的图层（Layer）中并以不同的样式和颜色进行标注。在切换房价的物业类型或地价信息时用户只能看到相应的图层。

其中的地图文件在数据库服务器和用户的本地地图文件中均有保存。

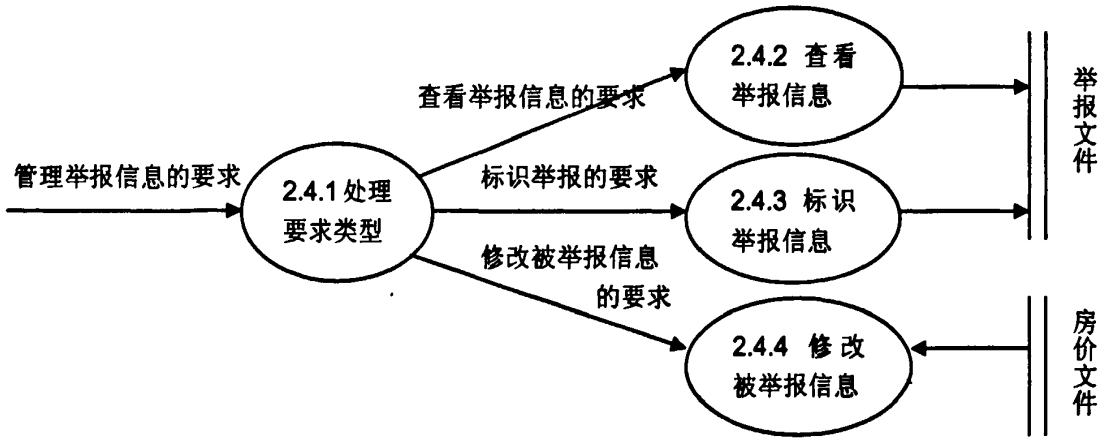

图 2-12 管理举报信息的二层数据流图

图 2-12 描述了管理员管理举报信息的要求。当软件的一般用户在使用软件的时候，如果发现房价信息有问题，可以按物业类型将楼盘的名称、具体问题上报给系统管理员。被举报的信息将统一由系统管理员进行确认并修订。

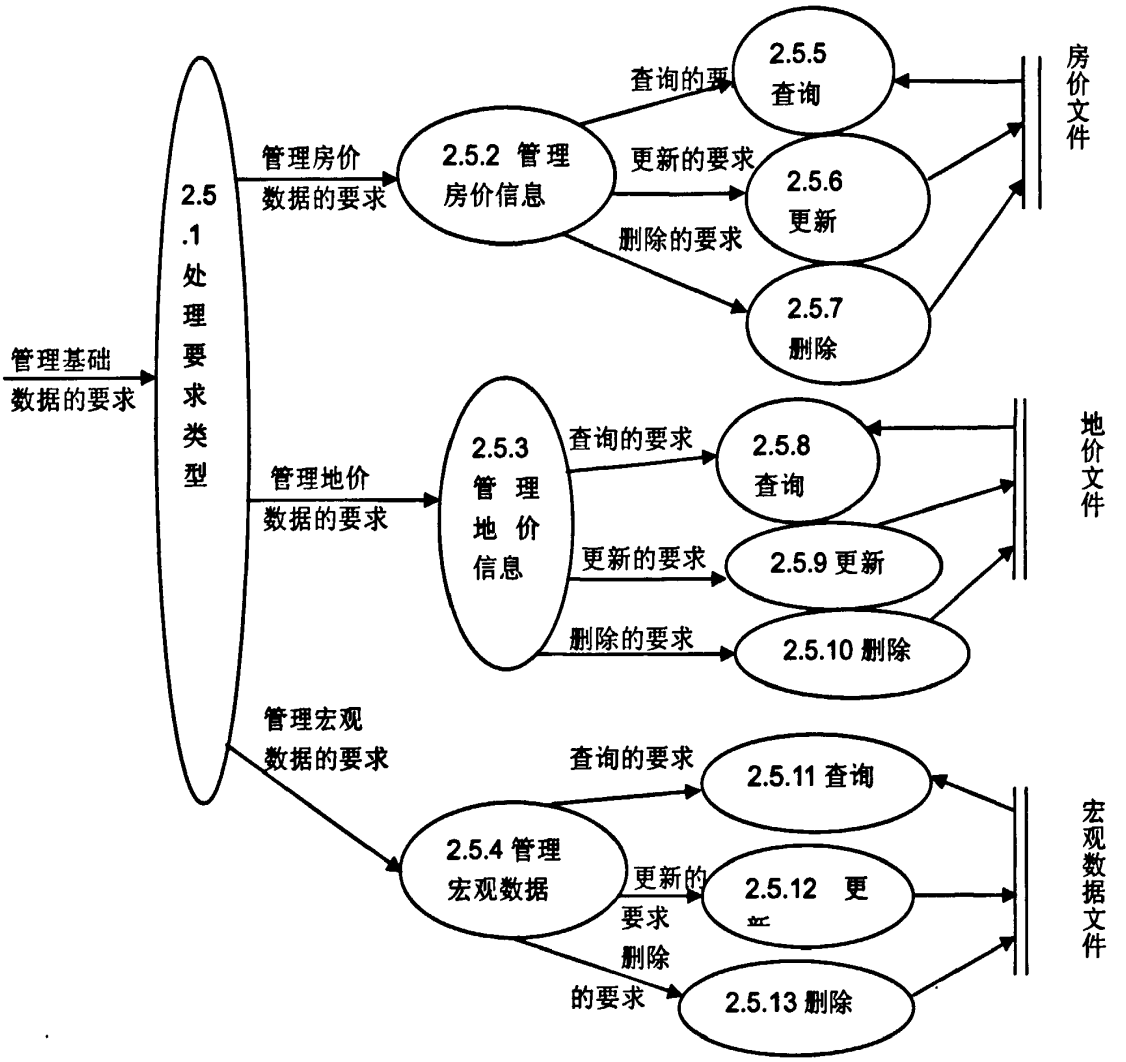

图 2-13 管理基础数据的二层数据流图

图 2-13 描述了管理员管理房价信息、地价信息、宏观数据的要求。其中房价信息按物业类型分为四类（住宅、商业、写字楼、工业），另外它还包括市调信息。地价信息又分为协议出让、招拍挂两种。所有这些数据构成了软件最核心的数据。他们都是由各个区域的信息采集员将统计的 Excel 文档上传至网站，系统管理员再将符合要求的数据导入数据库中。房价和地价的更新在程序的客户端完成，宏观数据的更新则在网站中实现。

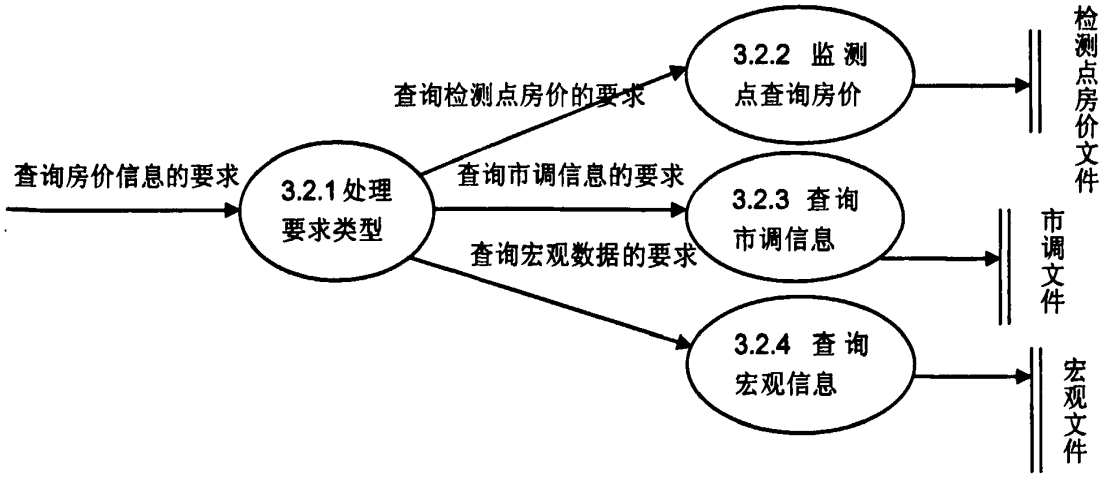

图 2-14 查询房价的二层数据流图

图 2-14 描述了系统中各种用户对房价信息（检测点房价信息、市调信息、宏观信息）的查询要求。所有的查询操作在软件的客户端实现。每个查询的结果都按照事先规定好的模板格式来显示信息。用户还可以根据自己的需要改变模板中的列名、排列顺序、显示的格式以及是否显示某个字段。用户修改模板的格式不会影响到其他用户。

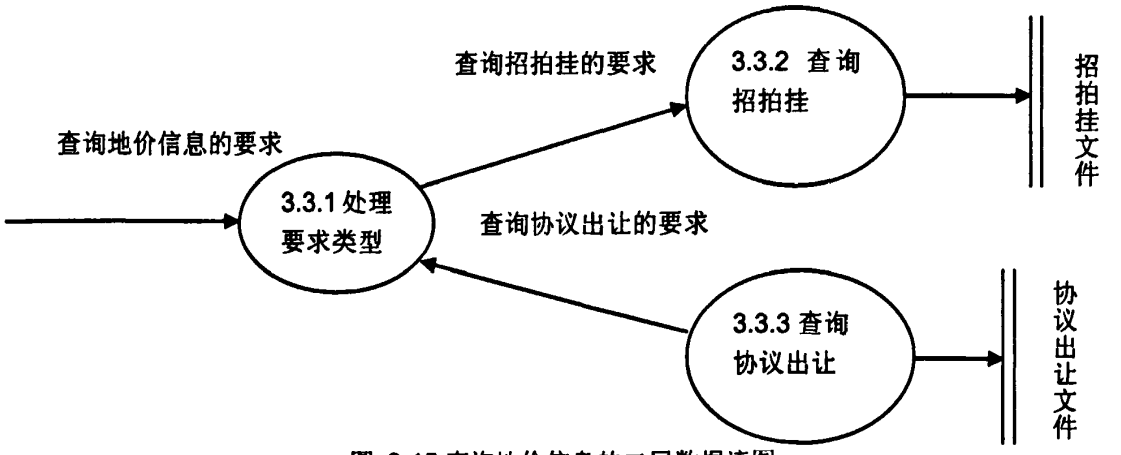

图 2-15 查询地价信息的二层数据流图

图 2-15 描述了系统中各种用户对地价信息（招拍挂信息、协议出让信息）的查询要求。和房价信息一样，用户可以自定义模板来改变信息的显示格式。

2.5 数据字典

数据存储名：用户文件

描　　述：存储软件使用人员的基本信息

数 据 组成：{用户名称+用户角色+访问次数+最后一次的访问时间+房价或地价用户标识}

数据存储名：地图文件

描　　述：存储电子题图的图层、点位信息

数 据 组成：{图层名称+图层显示顺序+点位编号+点位名称+点位横坐标值+点位纵坐标值+点位的添加时间+点位的项目编号+点位的项目名}

数据存储名：举报信息文件

描　　述：存储用户举报的信息

数 据 组成：{举报人+被举报房价信息的物业类型+被举报信息的项目名称+举报描述+举报日期+是否已经被处理+处理人}

数据存储名：基础数据文件

描　　述：基础数据文件由多个子文件组成

数 据 组成：{房价文件|地价文件}

数据存储名：房价文件

描　　述：房价文件由多个子文件组成

数 据 组成：{监测点房价文件|市调信息文件|宏观数据文件}

数据存储名：地价文件

描　　述：地价文件由多个子文件组成

数 据 组成：{招拍挂信息文件|协议出让信息文件}

数据存储名：市调信息文件

描　　述：存储市场调查信息的明细

数 据 组成：{项目名称+所在区县+具体位置+土地面积(平方米)+建筑面积(平方米)+售价(元/平方米)+租金（含物业、不含水电）+交易日期+用途+信息来源+联系电话+环线+备注+建筑年代+装修状况+物业费+楼层+朝向+楼型+主力户型及面积+车位状况及价格+调查人+项目流水号}

数据存储名：文档上传记录文件

描　　述：存储文档上传的信息

数 据 组成：{上传人+上传日期+文件名称+文件大小+文件类型+保存路径+是否已经导入数据库}

数据存储名：监测点房价文件

描　　述：存储监测点房价信息的明细，由 4 种类型的文件组成

数 据 组成：{住宅文件|写字楼文件|工业开发区文件|商业文件}

数据存储名：住宅文件

描　　述：存储监测点住宅信息的明细

数 据 组成：{项目编号+项目名称+具体位置+物业类型(普通住宅/公寓/别墅)+建筑规模(m2)+销售均价(元/m2)+容积率+楼型（塔楼/板楼/独栋别墅/联排别墅/双拼别墅/叠拼别墅）+建筑结构+总层数+装修状况+主力户型及面积+开盘时间+开盘价格(元/m2)+完工时间(建成年代)+预售成交面积(m2)+销售率+信息来源+联系电话+调查日期+供应类型(新增/在售新房/二手房)+备注}

数据存储名：写字楼文件

描　　述：存储监测点写字楼信息的明细

数 据 组成：{项目编号+项目名称+具体位置+物业类型(工业立项/住宅立项/纯写字楼)+建筑规模(m2)+销售价格(元/m2)+租金(含物业、不含水电,元/天. m2)+物业费(元/月.m2)+装修状况+层高+总层数+主力户型面积+完工时间(建成年代)+预售成交面积(m2)+信息来源+联系电话+调查日期+是否为新增供应+备注}

数据存储名：商业文件

描　　述：存储监测点商业楼盘信息的明细

数 据 组成：{项目编号+项目名称+具体位置+物业类型(底商/商业楼)+建筑规模(m2)+销售价格(元/ m2)+租金(含物业、不含水电,元/天. m2)+物业费(元/月.m2)+装修状况+层高+所在楼层+主力户型面积+完工时间(建成年代)+预售成交面积(m2)+信息来源+联系电话+调查日期+是否为新增供应+备注}

数据存储名：招拍挂文件

描　　述：存储地价招拍挂文件信息的明细

数 据 组成：{序号+交易文件编号+项目名称+宗地位置+所在区县+土地用途+环线位置+土地总面积(平方米)+建设用地面积+代征地面积+规划建筑面积(平方米)+交易方式+规划用途+竞价开始日期+招标、挂牌总价(万元)+招标、挂牌单价(元/平方米)+容积率+运作中心+土地状况+成交日期+成交价(万元)+受让单位+溢价幅度+备注+一级开发单位+土地级别+成交单价(元/平方米)}

数据存储名：工业开发区文件

描　　述：存储监测点工业开发区信息的明细

数 据 组成：{项目编号+开发区名称+具体位置+土地级别+占地规模+主营行业+基础设施(通平情况)+熟地价+土地取得费(征地费等)+土地出让金+标准厂房情况（结构/层数）+标准厂房情况（租金）+标准厂房情况（售价）+优惠政策+信息来源+联系电话+调查日期+备注}

数据存储名：协议出让文件

描　　述：存储地价协议出让信息的明细

数 据 组成：{序号+项目名称+宗地位置+委估单位+评估机构+所在区县+区县排序+用地位置描述+土地用途+用途排序+审定设计方案通知书号+建设用地面积(平方米)+地上建筑面积(平方米)+总建筑面积(平方米)+地上容积率+收益容积率+总容积率+地价区类/土地级别+评估熟地地面(元/平米)+评估熟地楼面(元/平米)+评估毛地地面(元/平米)+评估毛地楼面(元/平米)+评估土地出让金地面(元/平米)+评估土地出让金楼面(元/平米)+评审办建议价格毛地地面(元/平米)+评审办建议价格毛地楼面(元/平米)+评审办建议价格地面出让金(元/平米)+评审办建议价格楼面出让金(元/平米)+评委会审核价格毛地地面(元/平米)+评委会审核价格毛地楼面(元/平米)+评委会审核价格地面出让金(元/平米)+评委会审核价格楼面出让金(元/平米)+审定毛地地面价+审定毛地楼面价+审定地面出让金+审定楼面出让金+审定日期}

第 3 章 数据库设计

3.1 数据库概念结构设计

从用户的角度看待数据及处理要求和约束，产生一个反映用户观点的概念模式。概念模式降低了设计的复杂度，不受特定的 DBMS 的限制，避免了过早进入技术细节的考虑。同时概念模式也更容易让用户了解。

本文采用 ER 图方法进行数据库概念设计。

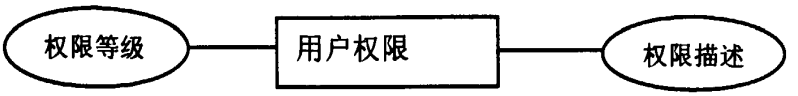

图 3-1 用户权限实体图

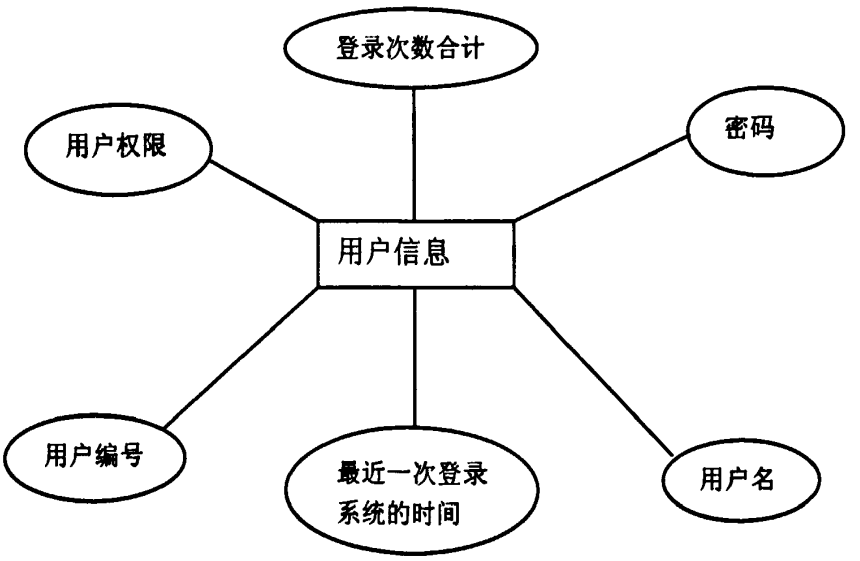

图 3-2 用户信息实体图

图 3-1、图 3-2 分别描述了权限实体和用户实体。用户和权限的联系是 N：1 的。

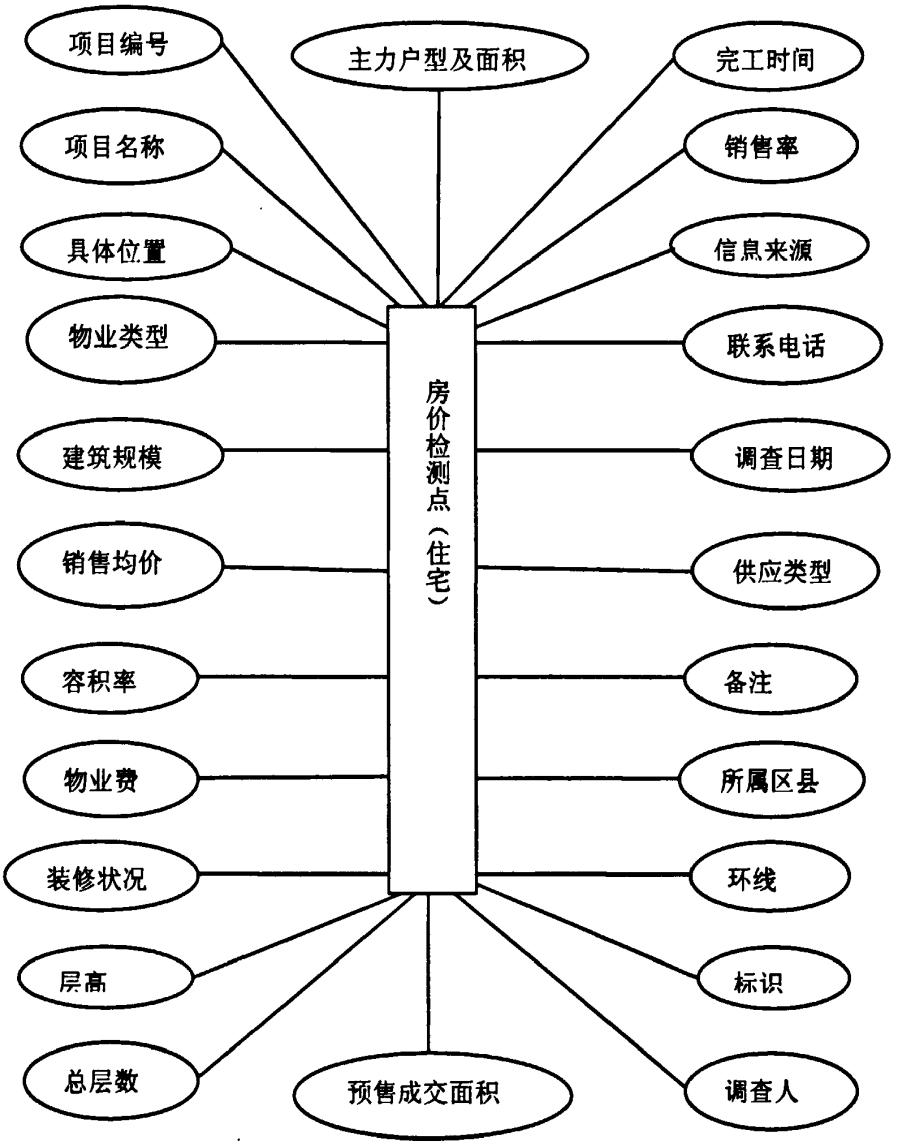

图 3-3 房价检测点实体图

房价检测点住宅、商业、写字楼、工业内容较为相似且属性过多，限于篇幅不再累述。地价信息中的招拍挂、协议出让两部分也存在同样问题，也不再一一罗列。

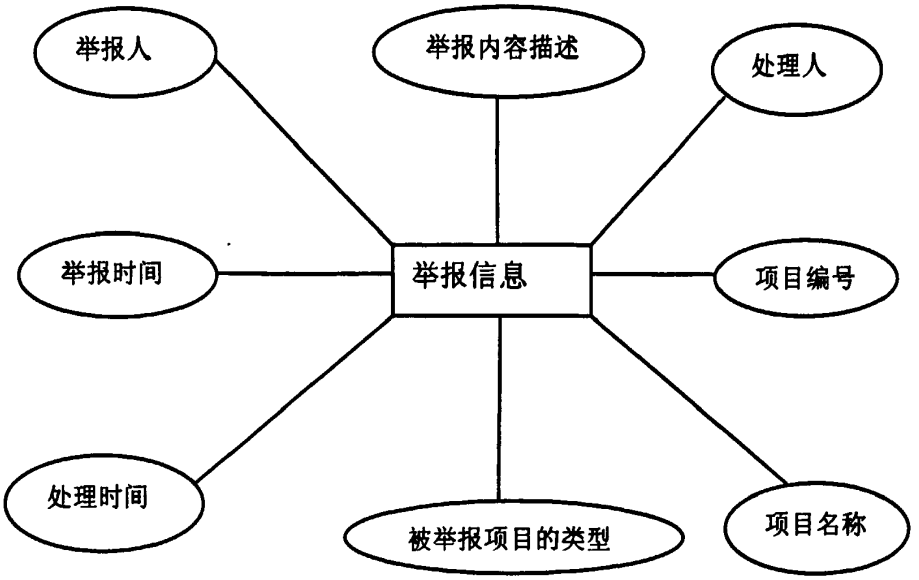

图 3-4 举报信息实体图

图 3-3、图 3-4 分别描述了房价监测点（住宅）实体和举报信息的实体。举报信息用来记录房价检测点住宅、商业、写字楼的项目编号等信息，实现举报信息的可追溯性。举报信息实体和住宅实体、商业实体、写字楼实体的关系均为 N:1。

3.2 数据库逻辑结构设计

数据库逻辑结构设计的目的是把概念设计阶段设计好的 E-R 图转换为与选用的 DBMS 所支持的数据模型相符合的逻辑结构。本软件采用的是关系型数据库 SQL Server 2000。把 E-R 模型转换为关系模型的逻辑结构如下。

表 3-1 地价协议出让信息表

地价协议出让信息表	Land				
字段名	数据类型	长度	主键	是否为空	说明
LandID	Int		√		IDENTITY(1, 1)
SerialNo	Nvarchar	20			项目编号
ProjectName	Nvarchar	200			项目名称
AddressDetail	Nvarchar	100		√	宗地位置
EvaluateUnit	Nvarchar	100		√	委估单位
EvaluateInstitution	Nvarchar	100		√	评估机构
County	Nvarchar	20		√	所在区县
CountyOrder	Nvarchar	100		√	区县排序
LandAddressDetail	Nvarchar	100		√	用地位置描述
LandPurpose	Nvarchar	100		√	土地用途
PurposeOrder	Nvarchar	100		√	用途排序
AuthorizedDesignId	Nvarchar	100		√	审定设计方案通知书号
LandArea	numeric	9		√	建设用地面积
BuildingArea	numeric	9		√	地上建筑面积(平方米)
TotalArea	numeric	9		√	总建筑面积(平方米)
LandTotalRate	numeric	9		√	总容积率
OverLandTotalRate	numeric	9		√	地上容积率
IncomeRate	numeric	9		√	收益容积率
LandPriceType	int	8		√	地价区类
EvaluateLandPrcie	numeric	9		√	评估熟地地面(元/平米)
EvaluateBuidingPrcie	numeric	9		√	评估熟地楼面(元/平米)
EvaluateGrossLandPrcie	numeric	9		√	评估毛地地面(元/平米)
EvaluateGrossBuildingPrcie	numeric	9		√	评估毛地楼面(元/平米)
EvaluateLandTransferPrice	numeric	9		√	评估土地出让金地面(元/平米)
EvaluateBuildingTransferPrice	numeric	9		√	评估土地出让金楼面(元/平米)
LabAdviceGrossLandPrcie	numeric	9		√	评审办建议价格毛地地面(元/平米)
LabAdviceGrossBuildingPrcie	numeric	9		√	评审办建议价格毛地楼面(元/平米)
LabAdviceLandPrcie	numeric	9		√	评审办建议价格地面出让金(元/平米)
LabAdviceBuildingPrcie	numeric	9		√	评审办建议价格楼面出让金(元/平米)
JuryAuthorizedGrossLandPrcie	numeric	9		√	评委会审核价格毛地地面（元/平米）
JuryAuthorizedGrossBuildingPrcie	numeric	9		√	评委会审核价格毛地楼面（元/平米）

JuryAuthorizedLandPrcie	numeric	9		√	评委会审核价格地面出让金（元/平米）
JuryAuthorizedBuildingPrcie	numeric	9		√	评委会审核价格楼面出让金（元/平米）
AuthorizedGrossLandPrcie	numeric	9		√	审定毛地地面价
AuthorizedGrossBuildingPrcie	numeric	9		√	审定毛地楼面价
AuthorizedLandTransferFee	numeric	9		√	审定地面出让金
AuthorizedBuildingTransferFee	numeric	9		√	审定楼面出让金
AuthorizedDate	Nvarchar	20		√	审定日期

说明：限于篇幅，省略地价招拍挂信息表。

表 3-2 房价监测点调查表(住宅)

房价-监测点调查表(住宅)	House				
字段名	数据类型	长度	主键	是否为空	说明
HouID	Int		√		IDENTITY(1,1)
SerialNo	Nvarchar	20			项目编号
ProjectName	Nvarchar	200			项目名称
AddressDetails	Nvarchar	1000		√	具体位置
EstateType	Nvarchar	200		√	物业类型
BuildingSize	numeric	9		√	建筑规模(m2)
BuildingPrice	numeric	9		√	销售均价(元/ m2)
BuildingRate	numeric	9		√	容积率
BuildingType	Nvarchar	100		√	楼型
BuildingStruct	Nvarchar	100		√	建筑结构
FloorCount	Nvarchar	100		√	总层数
Fitment	Nvarchar	100		√	装修状况
MainHouseSize	Nvarchar	1000		√	主力户型及面积
OpenDate	Nvarchar	100		√	开盘时间
OpenPrice	Nvarchar	100		√	开盘价格(元/ m2)
FinishDate	Nvarchar	100		√	完工时间(建成年代)
PresellArea	numeric	9		√	预售成交面积(m2)
SellRate	Nvarchar	100		√	销售率
InfoSource	Nvarchar	100		√	信息来源
Telephone	Nvarchar	100		√	联系电话
SurveyDate	Nvarchar	20			调查日期
Comment	Nvarchar	100		√	备注
County	Nvarchar	100		√	所属区县
RingRoad	Nvarchar	10		√	环线
bFlag	Bit	1			标识
People	Nvarchar	20			调查人

大型楼盘的开盘时间、开盘价格、完工时间会出现如下情况：[住宅为 2002

年，联排和独栋为 2004 年]、[联排 5500 元/平方米 叠拼 4800 元/平方米 双拼 6000 元/平方米]、[二期公寓毛坯 2009 年 6 月 30 日 二期公寓精装修 2008 年 12 月 30 日]。因此这些字段只能使用字符数据类型。

说明：限于篇幅，省略房价监测点调查表(商业)、房价监测点调查表(写字楼)、房价监测点调查表(工业开发区)。

以上为房地产信息管理系统的基础数据，分为房价信息和地价信息两部分。他们是用户最关心、最常使用的数据。

表 3-3 举报信息表

举报信息表	ReportInfo				
字段名	数据类型	长度	主键	是否为空	说明
ReportID	Int		√		IDENTITY(1,1)
ReportPeople	Nvarchar	20			举报人
ReportContent	Nvarchar	1000			举报内容描述
ReportDate	Smalldatetime				举报时间
ReportType	Nvarchar	10			被举报项目的类型（住宅、商业、写字楼、工业）
ProjectId	Nvarchar	40			项目编号
ProjectName	Nvarchar	200			项目名称
DealPeople	Nvarchar	20		√	处理人
DealDate	Smalldatetime			√	处理时间

举报信息表记录了被举报房价（包含住宅、商业、写字楼、工业 4 种类型）的项目编号和项目名称，他们作为本表的外键不能为空。

表 3-4 用户信息表

用户信息表	Users				
字段名	数据类型	长度	主键	是否为空	说明
Userid	Int		√		IDENTITY(1, 1)
UserNo	Nvarchar	20			用户编号
UserName	Nvarchar	20			用户名
UserPassword	Nvarchar	20			密码
Userights	Int				用户权限
loginCount	Int				登录次数合计 默认值 0
loginTime	smalldatetime				最近一次登录系统的时间

表 3-5 用户权限表

用户权限表	Rights				
字段名	数据类型	长度	主键	是否为空	说明
RightsId	Int		√		IDENTITY(1,1)
RightLevel	Nvarchar	20			权限等级（系统管理员、一般用户、高级用户）
RightDetail	Nvarchar	100		√	权限描述

用户信息表中的用户权限字段作为外键对应用户权限表的自增主键。两个表一起描述了用户及用户权限的信息。

第4章 服务器端系统的实现

4.1 开发工具的选择

微软的Visual studio 2005的发布代表了.Net计划进一步的成熟。Visual Studio 2005继承了Visual Basic6.0的易用性和可视化的集成开发环境。它使开发者可以在短时间内构造出满足用户需求的应用程序或者构建一个功能丰富的网站程序。Visual Studio2005集成了Winform程序开发（传统的桌面程序）和Asp.Net网页程序开发，控件式的开发模式让用户可以很快的使用并熟悉它。

鉴于客户实际的需求，本软件分成客户端程序、网站程序、Web服务三个模块。采用Visual Studio2005集成环境来开发是非常不错的选择。而后台数据库则选用微软的关系型数据库SQL Server2000。本软件的开发使用到了SQL Server的各种特性：Check 约束实现数据的完整性、OpenDataSource 读取本地 Excel 文件并将其导入SQL Server数据库、临时表、存储过程、事务等。

软件的网站部分使用了 Ajax 技术。这个当前最热门的技术，在 Visual studio 2005里得到了完整的支持。程序员可以使用Ajax Extensions1.0软件包中的控件，设置Update Panel中更新的触发条件，来快速的实现Ajax页面；也可以使用*.ashx文件来响应来自页面的请求并返回XML格式的数据流来更新页面中需要更新的那部分数据。

关于软件的Web Service部分，采用Visual studio 2005来开发也是非常快捷的。程序员只需要在*.asmx文件的隐藏代码（Code Behind）文件中实现继承Web Service的类即可，在需要对外公开的方法前声明属性WebMethod。

4.2 Web Service

客户端程序以及第三方软件都通过Web Service来访问数据库。本软件简化了客户端所使用Web Service的开发，仅仅是将访问数据库的两个必须要使用的方法（ExecuteNonQuery、ExecuteDataset）进行了封装。代码如下所示：

```
using System;
using System.Web;
using System.Web.Services;
```

```
using System.Web.Services.Protocols;
using System.Data;

[WebService(Namespace = "http://microsoft.com/ShouJiawebservices/")]
[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
public class Service : System.Web.Services.WebService
{
    public Service () { }
    //将公有方法声明为WebMethod对外开放
[WebMethod]
//执行存储过程、sql语句并返回结果集
    public DataSet getDataSet(string sql)
    {
        return SqlHelper.ExecuteDataset(SqlHelper.Conn,
CommandType.Text, sql);
    }

[WebMethod]
//执行存储过程、sql语句并返回最后一条语句所影响的行数
    public int getExecuteNonQuery(string sql)
    {
        return SqlHelper.ExecuteNonQuery(SqlHelper.Conn,
CommandType.Text, sql);
    }
}
```

这个简单的 Web Service 就这样实现了，它满足了软件访问数据库的基本要求。它允许用户执行各种 sql 命令：创建表、插入数据、删除数据，当然他还可以删除数据库！它没有对 sql 注入进行任何防范，随便一个 sql 注入工具都可以很轻易的把数据库毁掉！

为了能够正确使用、调试 Web Service 我们还要在 Web.Config 文件中声明

它所使用的协议。

```
<webServices>
    <protocols>
      <add name="HttpGet"/>
      <add name="HttpPost"/>
      <add name="HttpSoap"/>
      <add name="Documentation"/>
    </protocols>
</webServices>
```

考虑考第三方软件的对数据访问的要求，我们需要手动创建代理类。在 Visual Studio2005 中可以使用 wsdl.exe 命令行工具来实现，该工具需要一个 wsdl 文件作为输入。第三方接口的 Web Service 在 Service3rd.asmx 中实现，使用如下命令：

Wsdl http://localhost/ShouJiaWebService/Service3rd.asmx?wsdl。

Service3rd.asmx 将被编译成代理类，这样第三方程序就可以引用它了。

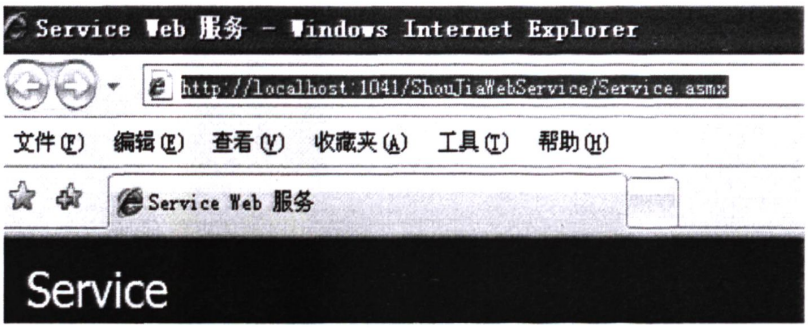

图 4-1 在 IE 中查看 Web Service 的接口

如图 4-1 所示，可以在 IE 浏览器中查看 Web Service 的接口列表。

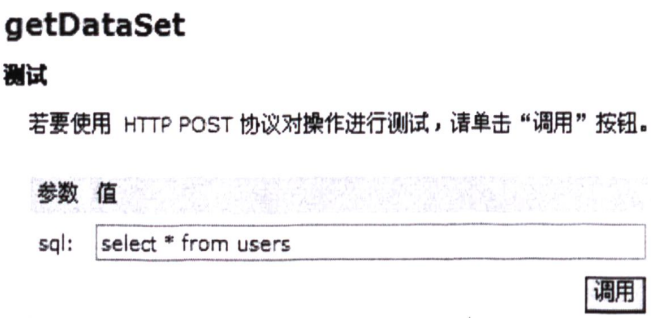

图 4-2 在 IE 中调用 Web Service 的接口

如图4-2所示，可以在IE浏览器中调用Web Service的方法。

```
- <NewDataSet xmlns="">
  - <Table diffgr:id="Table1" msdata:rowOrder="0">
      <Userid>1</Userid>
      <UserNo>10001</UserNo>
      <UserName>admin</UserName>
      <UserPassword>666888</UserPassword>
      <Userights>1</Userights>
      <loginCount>342</loginCount>
      <loginTime>2009-09-03T21:17:46.797+08:00</loginTime>
    </Table>
```

图4-3 在IE中调用Web Service的接口

如图4-3所示，Web Service的返回结果是标准的XML格式。而XML是一个标准，这使得返回的结果可以被任何可以解析XML的语言所处理。

如上所述，Web Service最大程度的减少了代码量，并提供很好的可扩展性。但是在实际应用中出现到了一些问题：由于Web Service基于SOAP（Simple Object Access Protocol）协议在网络间传递的是XML数据而非二进制流，这就造成在网络间（Internet）数据传输量较大，从而导致客户端响应明显延时，特别是在查询结果数据量较大或响应鼠标移动事件时显得尤为严重。在局域网内部并不存在这种现象。

解决响应延时的问题，有多种方案：

1.采用数据压缩技术。为了减少网络间的数据传输量，Web Service在每次发送查询结果前对数据进行压缩，客户端在接收前对数据进行解压。当数据量比较大时，即使采用数据压缩，依然会存在数据传输量大和响应延时的问题。

2.放弃Web Service，采用基于TCP/IP协议的Remoting技术。Remoting技术在网络间传递二进制数据流，同样也可以减少数据传输量。采用该办法程序要进行较大的调整，故不予采用。

3.使用SQL Server的复制技术。客户端程序的使用者分布在2个不同办公地点。把A地点的数据库服务器作为信息的发布者，B地点的数据库服务器作为信息的订阅者。数据的同步工作由SQL Agent服务自动完成，无需再编写额外的代码。客户端程序的使用者访问各自的数据库服务器，这样就使得所有用户查询的数据都在局域网内部传递，响应延时的问题迎刃而解。

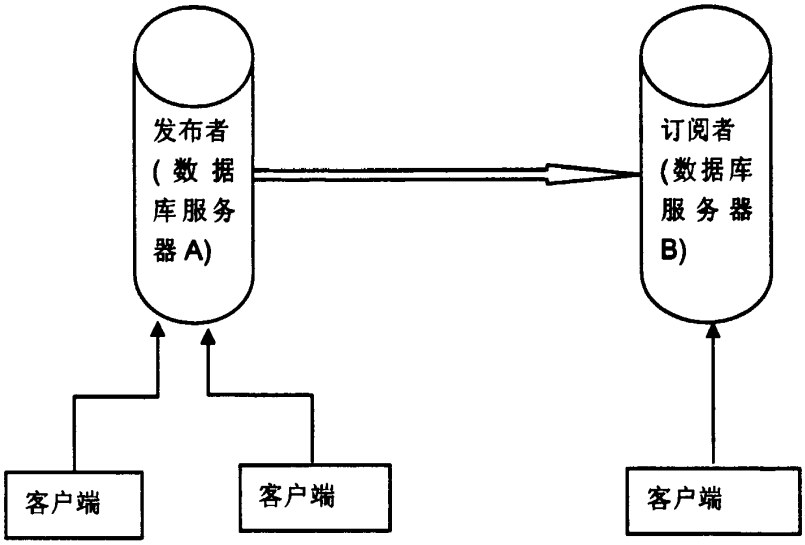

图 4-4 数据发布、订阅示意图

4.3 服务器端的安全性

为了确保 Web service 的安全性，我们采取平台/传输安全性和网关守卫两种方式进行客户端身份验证。

4.3.1 平台 / 传输安全性

图 4-5 显示了 ASP.NET Web 服务平台安全性体系结构。

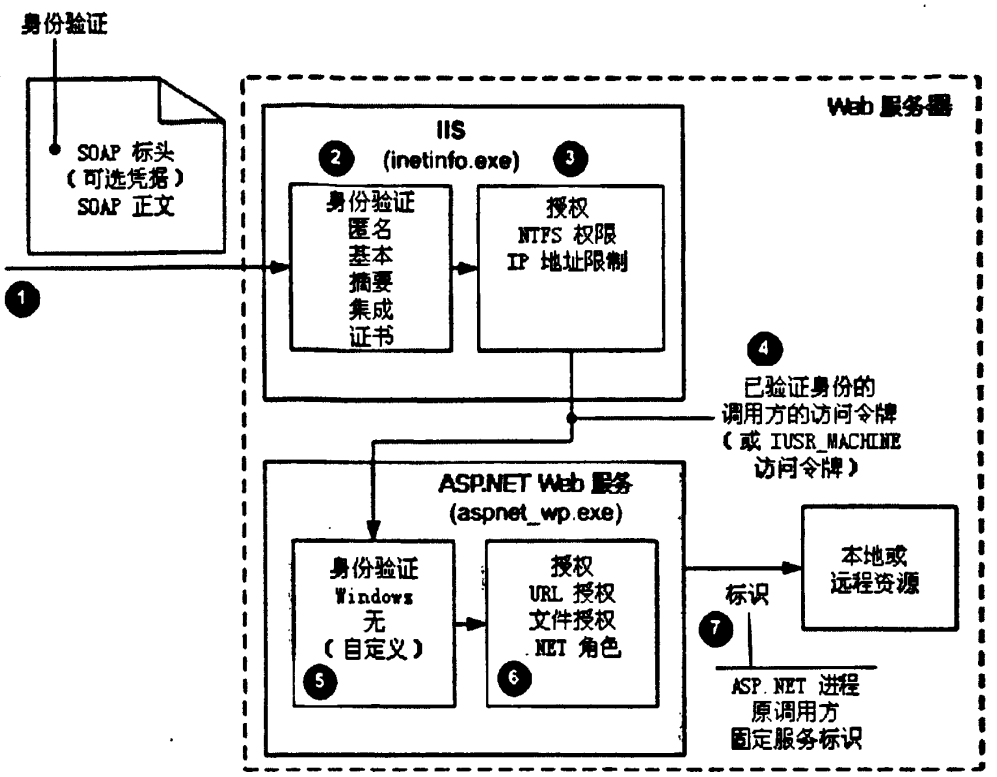

图 4-5 Web 服务安全性体系结构

图 4-5 阐释了 ASP.NET Web 服务提供的身份验证和授权机制。当客户端调用 Web 服务时，将按下列顺序激发身份验证和授权事件：

1. 接收到来自网络的 SOAP 请求。它是否包含身份验证凭证取决于所使用的身份验证类型。

2. IIS 可以有选择地使用基本、摘要、集成（NTLM 或 Kerberos）或证书身份验证对调用者进行身份验证。在不能使用 IIS（Windows）身份验证的异类环境中，可以将 IIS 配置为使用匿名身份验证。在这个方案中，可以使用消息级属性（例如，在 SOAP 标头中传递的票证）对客户端进行身份验证。

3. IIS 也可以配置为只接受来自特定 IP 地址的客户端计算机的请求。

4. IIS 将已验证的调用者的 Windows 访问令牌传递给 ASP.NET（如果将 Web 服务配置为使用匿名身份验证，则它可能是匿名 Internet 用户的访问令牌）。

5. ASP.NET 对该调用者进行身份验证。如果将 ASP.NET 配置为使用 Windows 身份验证，则此时不会进行任何其他的身份验证；IIS 对调用者进行身份验证。

如果使用的是非 Windows 身份验证方法，则将 ASP.NET 身份验证模式设置为“无”以使用自定义身份验证。

注 Web 服务目前不支持表单和 Passport 身份验证。

6. ASP.NET 通过使用 URL 授权和文件授权来授权访问所请求的 Web 服务（.asmx 文件），文件授权使用与 .asmx 文件关联的 NTFS 权限来确定是否将访问权限授予已验证身份的调用者。

注只能将文件授权用于 Windows 身份验证。

对于细分的授权，还可以使用 .NET 角色（以声明方式或编程方式）确保授权调用者访问所请求的 Web 方法。

7. Web 服务中的代码可以使用特定标识来访问本地和/或远程资源。在默认情况下，ASP.NET Web 服务不执行任何模拟，因此，配置的 ASP.NET 进程帐户提供该标识。也可以选择原调用者的标识或已配置的服务标识。

4.3.2 网关守卫

ASP.NET Web 服务中的网关守卫是：

- IIS
  - 如果禁用 IIS 匿名身份验证，则 IIS 只允许来自已通过身份验证的用户的请求。

- IP 地址限制

可以将 IIS 配置为只允许来自具有特定 IP 地址的计算机的请求。

•ASP.NET

- 文件授权 HTTP 模块（仅用于 Windows 身份验证）
- URL 授权 HTTP 模块

•主体权限要求和明确的角色检查

4.4 数据导入模块

分布在北京市各个区域的信息采集人员对所属区域的项目信息进行收集整理，并形成符合要求的Excel 数据。每个月定期将 Excel 表格上传至网站，然后由系统管理员将Excel 表格导入到服务器中的 SQL Server2000 数据库中。

为了确保导入数据的完整性，在定义表结构的时候使用了 Check 约束来限制调查日期、审定日期、成交日期、环线位置、调查人、所属区县等字段。由于客户有很多历史数据并不完全符合要求，所以在定义 Check 约束的时候使用关键字 NOCHECK 来指明不检查已有的数据。

以住宅数据表为例，具体代码如下：

```
//约束调查日期
ALTER TABLE House WITH CHECK ADD CONSTRAINT chk_SurveyDate CHECK (SurveyDate like '[2][0-9][0-9][0-9].[0-1][0-9].[0-3][0-9]')
//约束环线位置
ALTER TABLE House WITH NOCHECK ADD CONSTRAINT chk_ringroad CHECK (ringroad in ('二环内','二三环','三四环','四五环','五环外','远郊区'))
//约束调查人
ALTER TABLE House WITH NOCHECK ADD CONSTRAINT chk_people CHECK (len(people) >= 2)
//约束所属区县
ALTER TABLE House WITH NOCHECK ADD CONSTRAINT chk_county CHECK (len(county)>=2)
```

在 asp.net 的隐藏代码文件中，使用 SQL Server2000 提供的 OpenDataSource 方法读取已经上传至服务器中的 Excel 文件。使用 StringBuilder 对象拼写导入数据库的 sql 语句。由于在定义表的时候使用了 Check 约束，所以只有当 Excel 文件中的数据完全符合要求的时候才会导入成功（要么执行失败，一条也不会导入；要么执行成功，将数据全部导入）。

实现具体导入功能的代码见附件一。

在上述代码中使用了数据库的事务特性，用户对软件数据导入的要求分为 2 种方式：数据校验、数据导入。数据检验功能的实现并没有采用先将 Excel 的数据读到内存中，再逐项的去检查数据是否符合要求（这种方式将大大的增加代码量），而是利用数据库事务的特性显示的指明事务的开始、回滚。当插入语句执行成功时，代表数据是符合要求的，执行插入操作成功后再回滚事务以达到数据校验的目的；当插入语句扑捉到错误时，只需要将错误信息显示给用户即可。

图 4-5 为网站程序进行数据校验和数据导入的界面。

图 4-5 数据校验和数据导入的界面

4.5 图表统计模块

本网站需要统计各类举报信息的比例情况，要把 C#从后台数据库读出的统计数据作为前台页面 JavaScript 函数的参数，需要采用 Ajxa 的方式来实现。

具体的页面代码如下所示见附件二。

我们可以看到后台 C#代码将住宅、商业、写字楼被举报的总数拼写成一个

以逗号分隔的字符串，发送给页面的回调函数 callBack()。整个页面的执行过程：在加载页面的时候调用 createXMLHTTP()创建 XMLHTTP 对象；单击按钮后调用 GetData()发送 HTTP 请求给后台 c#文件 ReportStat.ashx；ReportStat.ashx 负责读取数据发回给页面的回调函数 callBcak()；回调函数调用 JavaScript 函数 StartDraw (paramSplitByComma)生成饼图。饼图的效果如图 4-6 所示：

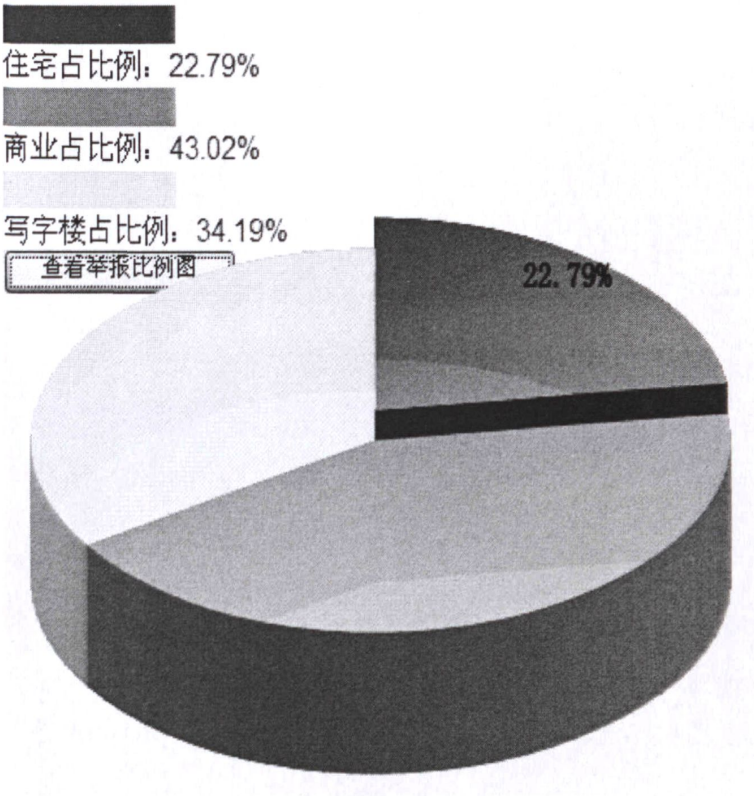

图 4-6 举报信息比例饼图

第 5 章 客户端程序的实现

5.1 区域房价检测点模块

区域房价检测点包含四个主要对象：住宅、商业、写字楼、工业开发区。我们在这里以住宅为例，实现如下功能：

1. 获得某个住宅的详细信息
2. 查询出满足条件的所有住宅信息的最新数据
3. 获得某个住宅所有的历史记录明细
4. 更新一条住宅信息
5. 删除一条住宅信息

如下代码类 CHouse 实现了上述的功能见附件三。

类 CHouse 中使用了 StringBuilder 类来处理字符串的连接。它是一种可变的字符串，每次长度不够用的时候，它都会自动的成倍增长。在构造频繁变化的字符串时，StringBuilder 类的效率将会比 String 对象高得多。

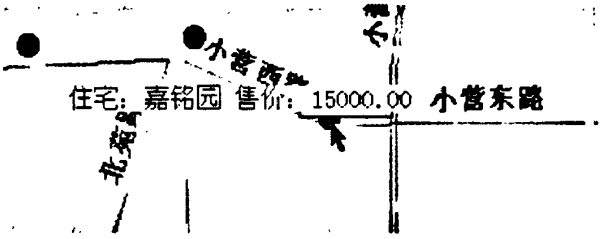

图 5-1 显示房价最新售价

如图 5-1 所示，当鼠标移动至某个地图点的时候，显示住宅最新的售价。代码如下所示：

```
//显示当前点的信息
private void axMap_Full_MouseMoveEvent(object sender,
AxMapXLib.CMapXEvents_MouseMoveEvent e)
{
    MapXLib.Features featrues;
    double mapX = 0; double mapY = 0;
     //当地图的放大值小于 45 的时候不显示信息
     if (axMap_Full.Zoom > 45) return;
     MapXLib.Point mouse = new MapXLib.Point();
     axMap_Full.ConvertCoord(ref e.x, ref e.y, ref mapX, ref mapY,
```

```
MapXLib.ConversionConstants.miScreenToMap);
mouse.Set(mapX, mapY);
//查找当前位置最上层的地图点信息
    features=m_searchLayer.SearchAtPoint(mouse,
MapXLib.SearchResultTypeConstants.miSearchResultTopmost);
if (featrues.Equals(null) || featrues.Count == 0)
{
        //lblShowInfo.Visible = false;
}
else
{
        StringBuilder sb = new StringBuilder();
        string sql = string.Empty;
          lblShowInfo.Text = string.Empty;
          lblShowInfo.Visible = true;
          foreach (MapXLib.Feature fea in featrues)
          {
              sb.Append("'" + fea.KeyValue.Trim() + "',");
          }
          if (sb.ToString().Length > 0)
          {
              sb.Remove(sb.Length - 1, 1);
          }
        //访问数据库，获得最新的房价信息
          sql = MapFeatrueType.GetInfoFangjia(sb.ToString());
          ShouJia.Service objService = new
MapInfoX.ShouJia.Service();
          ......
          DataSet ds = objService.getDataSet(sql);
          foreach (DataRow dr in ds.Tables[0].Rows)
          {
              lblShowInfo.Text = dr[0].ToString().Trim() + "： " +
           dr[2].ToString().Trim() + dr[3].ToString();
          }
           //显示房价信息
          lblShowInfo.Top = Convert.ToInt16(e.y) + 30;
          lblShowInfo.Left = Convert.ToInt16(e.x) + 120;
    }
}
```

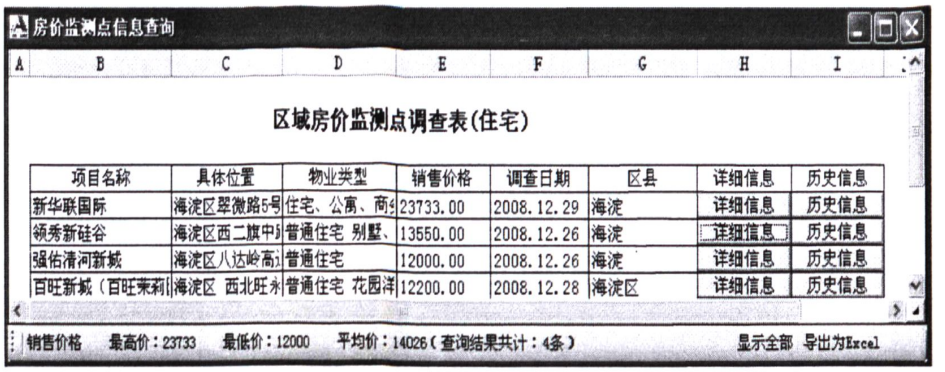

图 5-2 满足条件的住宅信息列表

图 5-2 为海淀区所有项目名称包含“新”字的住宅。这些信息是每个住宅最新的销售数据。用户还可以通过“详细信息”和“历史信息”查看楼盘的具体信息和所有的历史信息。

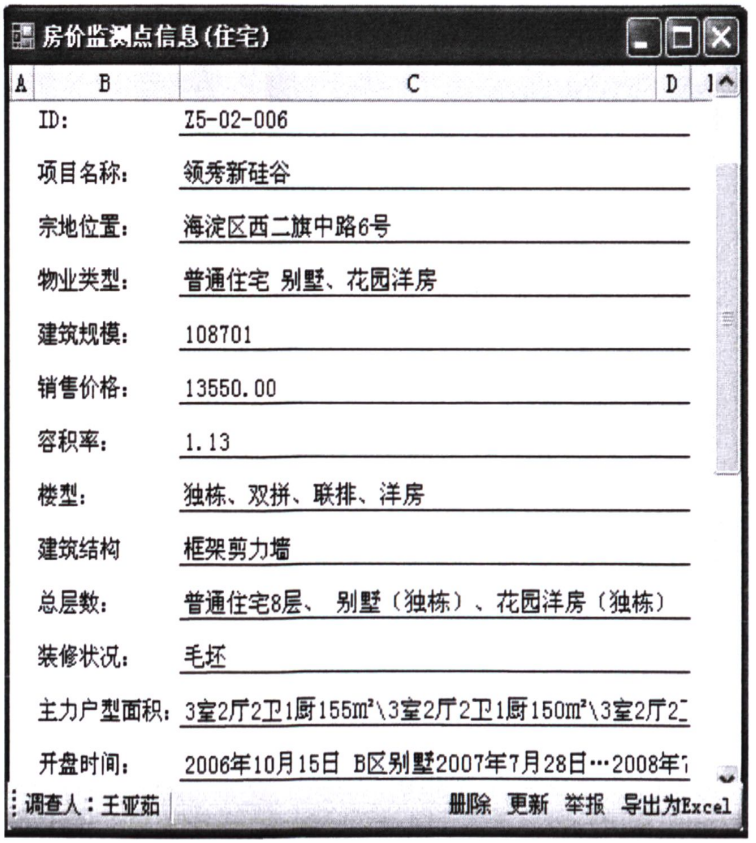

图 5-3 住宅最新的详细信息

图 5-3 显示了楼盘的最新的详细信息。管理员可以对信息进行删除、更新、导出 Excel 等操作。一般用户可以使用“举报”按钮对销售价格有问题的项目进行举报。

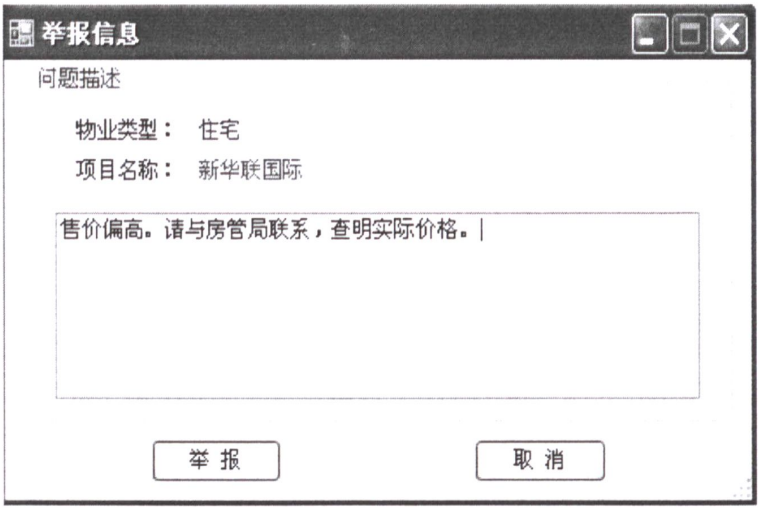

图 5-4 举报界面

图 5-4 为用户举报信息的窗口。被举报楼盘的编号、物业类型、项目名称、举报信息、举报人以及举报日期将会被记录到数据库中。被举报楼盘的编号作为外键关联到相应物业类型的数据表。

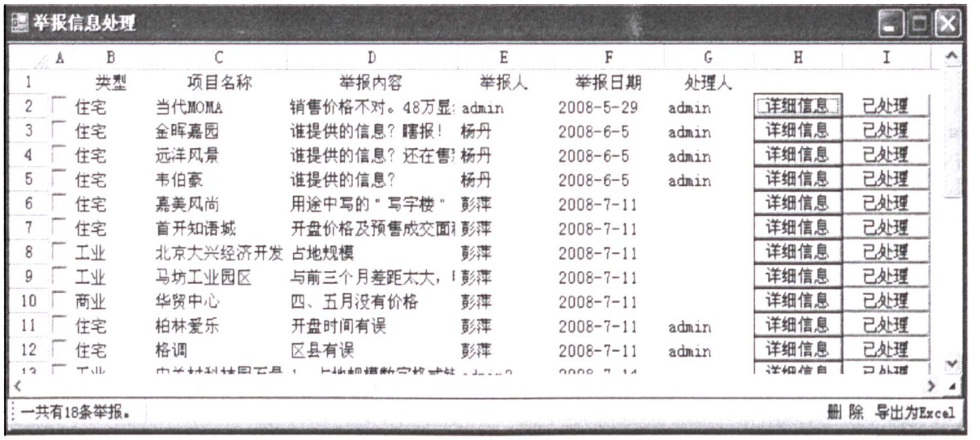

图 5-5 举报信息处理界面

处理举报信息的类，包含了如下功能：

1. 查询出所有的举报信息

2. 插入一条举报信息，记录被举报信息的主键和举报内容

3. 将一条举报信息标识为已处理

4. 删除一条举报信息

系统管理员可以通过举报信息处理界面（图 5-5 所示）查看被举报信息的详细情况，被举报信息经核实、处理之后标识为已处理状态。

5.2 地图更新模块

每个客户端都安装了电子地图，系统管理员会把新增加的项目信息添加到电子地图相应的图层中。当系统管理员更新电子地图的时候，将新增加的项目信息（所属图层、横坐标值、纵坐标值、项目编号、增加日期）记录到服务器的数据库中。其他客户端的使用者可以使用软件提供的“更新地图”功能来实现本地电子地图的更新。更新客户地图的时候有一个很重要的问题就是异常情况（比如掉电、死机）的处理，要保证每次更新地图的时候都要从上次失败的位置继续执行。我采取了先同步地图点，再写日志文件的方法来解决这个问题。程序流程图如下

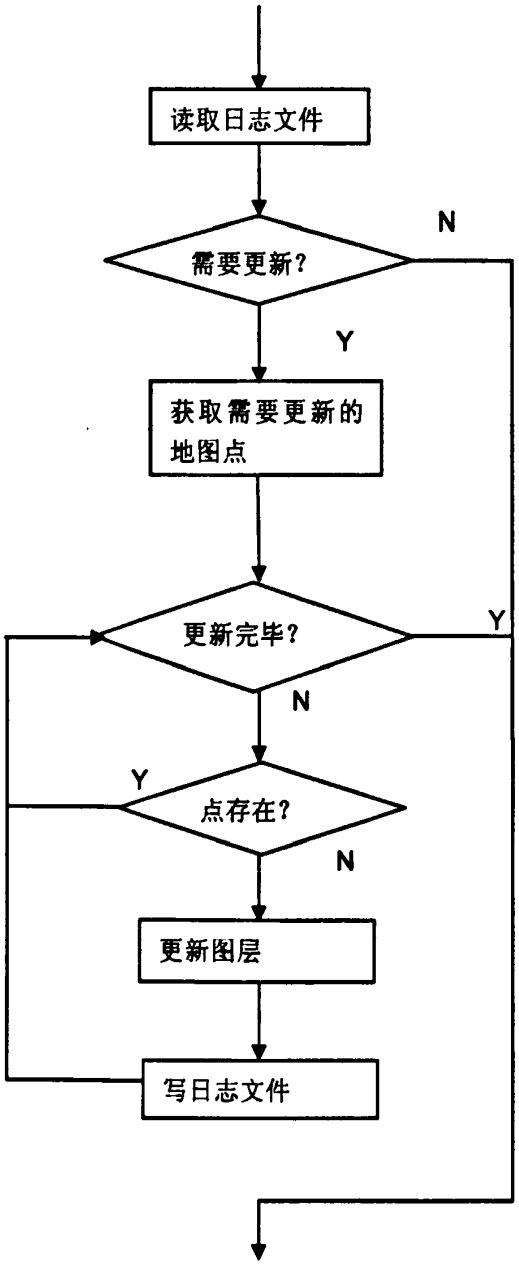

图 5-6 地图更新流程图

如图5-6中所示，客户端的日志文件记录了电子地图中每个可编辑图层的最后更新日期，服务器的数据库中记录了电子地图的更新记录。用户每次更新地图的时候，只从服务器取得大于日志更新日期的地图点。这样就可以确保地图更新的连续性。

但是这种方法也存在一些问题。由于更新地图在先，写日志文件在后，如果更新地图成功但是写日志文件失败(没有办法可以确保这两件事情具有事务性)，就会导致下次从服务器获取需要更新的点已经在地图中存在的问题。为了避免这种问题的产生，每更新一个点之前都要判断地图中是否已经存在该点。判断点位是否存在需要遍历整个地图，耗费的时间较多，有时会造成程序失去响应的假象。

更新地图的代码见附件四。

5.3 房价指数计算模块

为了能更好的反映房价的宏观走势，需要对历史数据进行统计分析，计算出销售均价、同比指数、环比指数、首佳指数并以折线图的形式直观地体现出来。各种指数的计算都是基于销售均价的。

用于计算销售均价、租金均价的过程要保存很多中间结果，所以采用存储过程来实现指数的计算。前台代码只需要调用相关的存储过程就可以获得最终的处理结果。住宅、商业、写字楼的销售均价以月或年为计算周期，可以按“所属区县”、“所属区域”、“所属环线”等条件来查看某个计算区间（最多计算24个月或24年）的销售均价、租金均价的信息。具体实现的代码见附件五。

上述存储过程sp_pricestate的参数@countrys、@regions、@ringroad分别代表区县、区域、环线，@daterange代表计算区间。这些参数都是以逗号分隔的字符串，比如sp_pricestate
'2008.06,2008.07,2008.08,2008.09,2008.10,2008.11,',',
'朝阳,海淀,昌平,朝阳,丰台,'。在存储过程中对这些字符串进行解析，使用嵌套循环分别处理区域字符串、计算区间字符串，在sql中解析字符串是很容易的事情，但是如果换成游标来实现循环，效率将会受到严重影响。存储过程是预编译的，相比SQL语句来说，它能获得更好的运行效率。采用存储过程还有一个特别的好处，如果客户的需求变化时，可以通过远程更新服务器的存储过程来改变存储过程实现的业务逻辑，这样就避免了更新客户端软件带来的麻烦。

另外，计算平均售价时，分母为0或空的情况要特别处理，防止出现除0的异常。

	2008.01	2008.02	2008.03	2008.04	2008.05	2008.06	2008.07	2008.08
定福庄	11745	10892	11745	11894	12051	12311	12679	11678
酒仙桥	12877	12644	12721	14698	15316	15364	17135	15586
望京	14143	14416	15863	15956	14897	13538	14180	14320
东八里庄	16006	16006	16006	17251	17344	18105	19459	16627
百子湾	18745	18745	16866	18845	19159	18007	20396	20580
东花市	23501	23651	22827	22710	23928	19602	21618	20803
三元桥	26525	26525	31370	31416	25909	34323	31170	31113

图5-7 按区域计算平均房价的结果

如图5-7所示，按“所属区域”计算定福庄、酒仙桥、望京、东八里庄、百子湾、东花市、三元桥的平均房价。

在销售均价的基础上计算环比指数、同比指数、首佳指数。其中，销售均价有两个计算方式：算数平均、加权平均。当销售价格或楼盘销售面积发生改变的时候，允许用户重新计算相应月份的销售均价及各种指数。

各种指数的计算公式如下。

环比 =(本月销售均价 － 上月销售均价)÷ 本月销售均价

同比 =(今年本月销售均价 － 去年本月销售均价)÷ 今年本月销售均价

首佳指数 =(本月平均销售均价 ÷ 2008年1月的销售均价)×100

实现各种指数计算的代码见附件六。

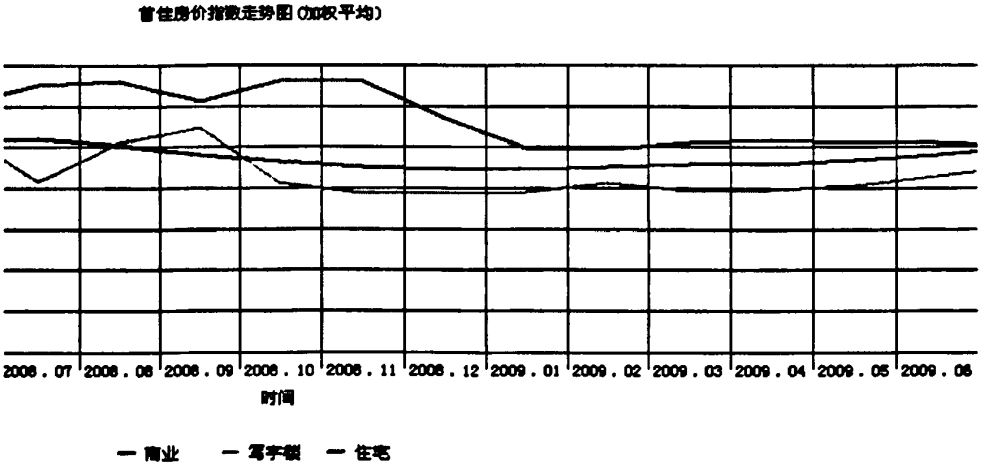

图5-8 商业、写字楼、住宅销售价格走势图

如图5-8所示，北京市2008年7月份到2009年6月份商业、写字楼、住宅的销售价格走势图。从蓝色的折线（左起中线）可以看出住宅的均价从2008年

7月开始呈现出下降的趋势，一直到2009年年初才开始回暖。而商业（左起上线）和写字楼（左起下线）的售价从2008年的10月底11月初开始大幅下跌。这些曲线真实的反映了北京房价变化的情况。

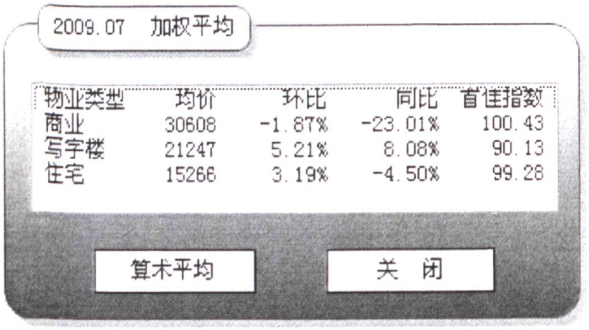

图5-9 2009年7月的指数

如图5-9所示，为2009年7月的环比、同比、首佳指数。

总结与展望

本论文所研究设计的房地产信息管理系统很好的实现了将房地产信息采集、查询、统计以及电子地图定位融为一体的软件。它在用户查询房价信息的同时，可以精确的在电子地图中显示出相应楼盘的地理位置。信息采集使用统一的模板，统一的数据格式，提高了数据的准确性、可靠性。用户对采集来的基础数据进行统计分析，可以按区域、环线、区县来统计房价的平均售价以及环比指数、同比指数、首佳指数。这些统计结果为用户分析房价波动和宏观经济因素之间的关系、判断未来房价的走势，提供了很好的参考价值。

在以后的工作学习当中还需要做到以下几点：

（1）增强系统的安全性研究及开发，保障系统应用安全，保障企业的数据安全。

（2）完善系统框架结构，增强其可扩展性，拓展其应用范围，增添更多功能。

致 谢

首先要感谢指导我完成论文写作的田怀文教授。在整个论文的写作过程中，从选题、大纲、内容组织、书写规范等各个方面，田教授都给予了悉心的指导，通过不断与指导老师勾通，不断改进程序，使其更加完美，方便用户使用，使系统更具特色。在此衷心地感谢田教授，谢谢您的指导和帮助！

在开发过程中，单位同事、同学在程序开发的技术方面给予我了很多支持，并提出了很多宝贵的意见。在这里对他们表示感谢！

在论文即将完成之际，我的心情非常激动。身边很多通过在职研究生考试取得成功的同事，他们一边工作一边学习，不仅在工作中取得了成绩，还在学习方面取得了进步，所有这一切都是大家努力和坚持学习的结果。感谢上级领导给我们这次机会，现要感谢所有为我们工程硕士班付出汗水和努力的局党干校工作人员，特别是要感谢我们的班主任刘老师！感谢所有帮助过我的老师，谢谢你们的无私奉献！

最后，我谨向百忙之中来参加论文答辩工作的所有老师致以衷心的感谢！

参考文献

[1] 王立福，麻志毅，张世琨. 软件工程（第二版）. 北京大学出版社，2002

[2] 丁宝康. 数据库原理. 经济科学出版社，2000

[3] John Papa，Matthew Shepker. SQL Server7 编程技术内幕. 机械工业出版社，2001

[4] Karli Wastson，Marco Bellinaso. C#入门经典. 清华大学出版社，2006

[5] Jesse Liberty，Dan Hurwitz. Programming ASP.NET. 电子工业出版社，2007

[6] Christian Nagel,Bill Evjen,Jay Glynn. C# 高级编程（第6版. 清华大学出版社,2008

[7] 郑华，房地产市场分析方法，电子工业出版社，2003

[8] 徐小平基于 Internet 的市场分析系统的设计与实现[期刊论文]-计算机工程与设计 2004(04)

[9] 叶艳兵;丁烈云房地产预警指标体系设计研究[期刊论文]-基建优化 2001(03)

[10] 张建;刘民;崔小青房地产项目的计算机辅助可行性研究 2004(12)

[11] 马智亮;邓子瑜;李恒房地产项目可行性研究辅助系统的建模[期刊论文]-计算机工程与应用 2005(01)

[12] 张志杰;陈龙乾房地产预警一般流程分析及要点诠释[期刊论文]-建筑经济 2004(08)

[13] 徐小平，基于 Internet 的市场分析系统的设计与实现，计算机工程与设计，2004

[14] 丁烈云，徐泽清，城市房地产预警系统的设计与开发，基建优化，2000

[15] 马智这，邓子瑜，房地产项目可行性研究辅助系统的建模，计算机工程与应用，2005

[16] 林锋 房地产中介公司管理信息系统的开发 南昌大学学报[J] 1996.3 第20卷第一期。

[17] 龚琪.Web Services 技术及其应用[J]. 计算机时代.

附录-源程序代码

附件一：

实现具体导入功能的代码如下：

```
protected void lbIn_Click(object sender, EventArgs e)
{
    string tableName =string.Empty;          //所指定导入 SQLSERVER
表名
    string showName = string.Empty;          //选择显示的信息
    string excelPath = lblInFilePath.Text.Trim();  //Excel 表在服务器上的
存储路径
    ……

    StringBuilder sb = new StringBuilder();
string sqlWhere = "";  //查询条件
//获得 Excel 文件在服务器的绝对路径
string path = Server.MapPath("~/") + "uploadfile\\" + excelPath;
   //如果操作是"数据校验"，那么显示声明数据库事务，在执行完导入
命令后再执行后回滚操作。借用数据库的事务，快捷的实现了数据校验
的功能。
    sb.Append("BEGIN TRAN checkExcel" + "\n");

    sb.Append(" INSERT INTO " + tableName);
    //错误提示信息
    StringBuilder sbTips = new StringBuilder();
    //读取住宅 Excel 的 sql 语句
    if (tableName == "House")    //住宅
    {
        int bflag = 0;
        if (showName == "房价监测(住宅)")
            bflag = 1;
        else if (showName == "市调信息(住宅)")
            bflag = 0;

      sb.Append("(SerialNo,ProjectName, AddressDetails, EstateType,
    BuildingSize, BuildingPrice," + "\n");
      sb.Append("BuildingRate, BuildingType, BuildingStruct,
    FloorCount, Fitment, MainHouseSize, " + "\n");
      sb.Append("OpenDate, OpenPrcie, FinishDate, PresellArea,
    SellRate, InfoSource, Telephone, " + "\n");
     sb.Append("SurveyDate, SupplyType, Comment, county,
              RingRoad,people, bFlag) " + "\n");
      sb.Append(" SELECT " + "\n");
```

```
        sb.Append("[序号],[项目名称],[具体位置],[物业类型(普通住宅/公寓
                  /别墅)],
                  [建筑规模(平方米)],convert(money,[销售价格(元/平方
                  米)])," + "\n");
        sb.Append("[容积率],[楼型(塔楼/板楼/独栋别墅/联排别墅/双拼别墅/
                  叠拼别墅)],[建筑结构]," + "\n");
        sb.Append("[总层数],[装修状况],[主力户型面积],[开盘时间],[开盘
                  价格
                  (元/平方米)],[完工时间(建成年代)],convert(money,[预售
                  成交面积(平方米)])," + "\n");
        sb.Append("[销售率],[信息来源],[联系电话],Replace([调查日
                  期],'. ','.'),
                  [供应类型(新增/在售新房/二手房)],[备注],[区县],[环线
                  位置],
                  [调查人]," + bflag);
            sqlWhere = " WHERE [项目名称] IS NOT NULL ";
        }
        //INSERT INTO ... SELECT 语句拼写完毕
        sb.Append(" FROM OPENDATASOURCE" + "\n");
        sb.Append("('Microsoft.Jet.OLEDB.4.0','Data Source=\"" + path +
"\";User ID=Admin;Password=;Extended Properties="Excel 8.0; HDR=YES;
IMEX=1" ')...[Sheet1$]" + "\n");
        if (!sqlWhere.Equals(""))
        {
            sb.Append(sqlWhere);
        }

        //如果是"数据校验"，则回滚导入操作
    if (m_bIsCheck == true)
        {
            sb.Append("\n" + "ROLLBACK TRAN checkExcel");
        }
        else
        {
            sb.Append("\n" + "COMMIT TRAN checkExcel");
        }

        DbHelperSQL objHelper = new DbHelperSQL();
        string errMessage = string.Empty;                    //扑捉 sql 错误
        StringBuilder errShow = new StringBuilder();         //错误显示
        //执行 sql 命令，并且获取错误信息
        bool bSuccess = objHelper.SqlExecute(sb.ToString(),ref errMessage);
        //数据检验或导入成功
        if (bSuccess)
```

```
        {
            //导入成功，将文件标识为已导入
        ……
        }
        else
        {
            //显示并处理错误信息
           ……
        }
}
```

附件二：

```
<head runat="server">
    //引入画饼图的 JavaScript 脚本文件 Pie3D.js
     <script type="text/javascript" src="js/Pie3D.js"
charset="gb2312"></script>
</head>

<form id="form1" runat="server">
    ......
</form>

<script type="text/javascript">
        function createXMLHTTP() {
            var xmlHttp = false;
            var arrSignatures = ["MSXML2.XMLHTTP.5.0",
"MSXML2.XMLHTTP.4.0",
                                 "MSXML2.XMLHTTP.3.0",
"MSXML2.XMLHTTP",
                                 "Microsoft.XMLHTTP"];
            for (var i = 0; i < arrSignatures.length; i++) {
                try {
                    xmlHttp = new ActiveXObject(arrSignatures[i]);
                    return xmlHttp;
                }
                catch (oError) {
                    xmlHttp = false;
                }
            }

            if (!xmlHttp && typeof XMLHttpRequest != 'undefined') {
                xmlHttp = new XMLHttpRequest();
            }
            return xmlHttp;
        }
        //加载页面的时候创建 XMLHTTP 对象
        var xmlReq = createXMLHTTP();

        // 发送 ajax 处理请求，并指明响应请求的.ashx 文件
        function GetData() {
            var url = "ReportStat.ashx"; //.ashx 文件
            xmlReq.open("get", url, true);
            xmlReq.setRequestHeader("If-Modified-Since", "0");
            xmlReq.onreadystatechange = callBack;
            xmlReq.send(url); // 发送文本
```

```
            }

            function callBack() {
                if (xmlReq.readyState == 4) {
                    if (xmlReq.status == 200) {
                        eval("StartDraw(xmlReq.responseText);");
                    }
                    else if (xmlReq.status == 404) {
                        alert("请求的网址不存在.");
                    } else if (xmlReq.status == 403) {
                        alert("访问拒绝.");
                    } else
                        alert("错误 " + xmlReq.status);
                }
            }
            //调用 JavaScript 的脚本画饼图
            function StartDraw(paramSplitByComma){
              ......
              pie.draw();
          }
    </script>
```

用 C#获取各种物业类型被举报数据的代码如下所示。

```
    public class ReportStat : IHttpHandler
    {
        public void ProcessRequest (HttpContext context) {
            context.Response.ClearContent();
            context.Response.ContentType = "text/plain";
            context.Response.Cache.SetCacheability(HttpCacheability.NoCache);
//无缓存
            //获取数据
            int nHouse = CReportInfo.GetStat("House");
    int nBusiness = CReportInfo.GetStat("Business");
    int nOffice = CReportInfo.GetStat("Office");

    string StatString= nHouse.ToString() + "," + nBusiness.ToString() + "," +
nOffice.ToString();
            context.Response.Write(StatString);
        }
    }
```

附件三：

```
public class CHouse
    {

        /// <summary>
        /// 1）获得某个住宅的详细信息
        /// </summary>
        /// <param name="key">住宅信息主键</param>
        /// <returns></returns>
        public static string getDetailsOneHouseInfo(string key)
        {
            ......
        }

        /// <summary>
        /// 2）查询出满足条件的所有住宅信息的最新数据
        /// </summary>
    public static string  getDetailsHouseInfo(string ProjectName,string
EstateSubType,string County,string AddressDetails,string Region, string SupplyType,
string SurveyDateB, string SurveyDateE, string RingRoad, string BuildingPrice)
        {
            StringBuilder sb = new StringBuilder();

    sb.Append(" SELECT  HouID,SerialNo , ProjectName, AddressDetails,
EstateType, BuildingSize, BuildingPrice, ");
    sb.Append(" BuildingRate, BuildingType, BuildingStruct, FloorCount, Fitment,
MainHouseSize, ");
    sb.Append(" OpenDate, OpenPrcie, FinishDate, PresellArea, SellRate,
InfoSource, Telephone, ");
            sb.Append(" SurveyDate, SupplyType, Comment, County,
RingRoad ,people ");
            sb.Append(" FROM House WHERE 1 = 1 ");

            if (!ProjectName.Equals(""))
                sb.Append(" AND ProjectName like '%" + ProjectName +
"%' ");
            if (!EstateSubType.Equals(""))//物业类型
        sb.Append(" AND EstateType like '%" + EstateSubType + "%' ");
            if (!County.Equals(""))
                sb.Append(" AND County like '%" + County + "%' ");
            if (!AddressDetails.Equals(""))
                sb.Append(" AND AddressDetails  like '%" +
AddressDetails + "%' ");
            if (!Region.Equals(""))
```

```
                {
    //所在区域，是住宅编号的前五位
                    sb.Append(" AND left(serialno,5) = '" +
Region.Substring(0,5) + "' ");
                }

                if (!SupplyType.Equals(""))
                    sb.Append(" AND SupplyType like '%" + SupplyType + "%'
");
                if (!BuildingPrice.Equals(""))
                    sb.Append(CComputerPrice.price("BuildingPrice",
BuildingPrice));//销售价格
                if (!RingRoad.Equals(""))
                    sb.Append(" AND RingRoad like '%" + RingRoad + "%' ");

                if (SurveyDateB.Equals("") && SurveyDateE.Equals("")) //默认
显示最新数据
                {
                    sb.Append(" and (houid in (select houid from house h1,");
    sb.Append(" (select serialno,max(surveydate) as max_surveydate from house
group by serialno) h2 ");
    sb.Append("   where h1.surveydate   = h2.max_surveydate and h1.serialno =
h2.serialno))");
                }
                else if (!SurveyDateB.Equals("") && !SurveyDateE.Equals(""))
    sb.Append(" AND SurveyDate BETWEEN '" + SurveyDateB + "' AND '" +
SurveyDateE + "' ");
                else
                {
                    if (!SurveyDateB.Equals(""))
                        sb.Append(" AND SurveyDate >= '" + SurveyDateB +
"' ");
                    else if (!SurveyDateE.Equals(""))
                        sb.Append(" AND SurveyDate <= '" + SurveyDateE +
"' ");
                }

                return sb.ToString();
            }

        /// <summary>
          ///3）获得某个住宅所有的历史记录明细
        /// </summary>
          public static string getHouseHistoryInfo(string ProjectName)
```

```
            {
                ......
            }
        /// </summary>
          ///4）更新一条住宅信息
        /// </summary>
    public static string updateSql(string v_HouID, string v_SerialNo, string
v_ProjectName, string v_AddressDetails, string v_EstateType, string v_BuildingSize,
string v_BuildingPrice, string v_BuildingRate, string v_BuildingType, string
v_BuildingStruct, string v_FloorCount, string v_Fitment, string v_MainHouseSize,
string v_OpenDate, string v_OpenPrcie, string v_FinishDate, string v_PresellArea,
string v_SellRate, string v_InfoSource, string v_Telephone, string v_SurveyDate,
string v_SupplyType, string v_Comment, string v_county, string v_RingRoad)
            {
                ......
            }
        /// </summary>
        ///5）删除一条住宅信息
    /// </summary>
            public static string deleteSql(string v_HouID)
            {
                ......
            }

    }
```

附件四：

更新地图的代码如下。

```
private void updateMaps()
{
    //日志文件记录的更新时间
     string lastUpdateTime = “”;
//逐个更新每个图层的点位
    //房价图层
    lastUpdateTime =
ConnectionStringClass.GetfileContent(“fangjia”,"lastTime.txt");
    UpdateMapFeatures.UpdateLayer(toolUpdateMapInfo, axMap_Full,
axMap_Full.Layers._Item("fangjia"), "fangjia", lastUpdateTime);
     //地价图层
    lastUpdateTime =
ConnectionStringClass.GetfileContent(“dijia”,"lastTime.txt");
    UpdateMapFeatures.UpdateLayer(toolUpdateMapInfo, axMap_Full,
axMap_Full.Layers._Item("dijia"), "dijia", lastUpdateTime);
     //招拍挂图层
    lastUpdateTime =
ConnectionStringClass.GetfileContent(“ZPG”,"lastTime.txt");
    UpdateMapFeatures.UpdateLayer(toolUpdateMapInfo, axMap_Full,
axMap_Full.Layers._Item("ZPG"), "ZPG", lastUpdateTime);

     MessageBox.Show("更新完毕！", "消息");
}
```

更新图层的代码如下。

```
public static void UpdateLayer(System.Windows.Forms.ToolStripStatusLabel
label , AxMapXLib.AxMap myMap, MapXLib.Layer mylyr,string mapType,string
lastUpdateTime)
{
     ……
    //读取图层新增加的点位
     sql = UpdateMapFeatures.select(lastUpdateTime, mapType);
     DataSet ds = objService.getDataSet(sql);

     if (ds.Tables.Count > 0 && ds.Tables[0].Rows.Count > 0)
     {
         label.Text = "准备更新地图" + mapType + "共有" +
           ds.Tables[0].Rows.Count.ToString() + " 个点，请您稍候。";

         foreach (DataRow dr in ds.Tables[0].Rows)
         {
             //判断该点是否已经存在
```

```
                    tempFeature = findFeatureByKeyValue(mylyr,
dr["mfSerialNo"].ToString());
                    if (tempFeature == null )
                    {
                        //不存在，则创建新的
    CreatPoint(myMap, mylyr, dr["mfSerialNo"].ToString(),
        Convert.ToDouble(dr["mfMapX"].ToString()),
    Convert.ToDouble(dr["mfMapY"].ToString()));
    //记录更新时间
    ConnectionStringClass.WritefileContent(mapType ,"lastTime.txt",
        DateTime.Now.ToString());
                    }
                }
            }
            else
            {
                label.Text = mapType + "图层已是最新，无需更新！上次更新日期：
" +
                    ConnectionStringClass.GetfileContent(mapType ,"lastTime.txt")
+ "。";
            }
        }

        /// <summary>
        /// 将新增点位写入图层
        /// </summary>
        /// <param name="myMap">所属地图</param>
        /// <param name="mylyr">所属图层</param>
        /// <param name="myKeyValue">点的编号</param>
        /// <param name="Mapx">地图坐标 X </param>
        /// <param name="MapY">地图坐标 Y </param>
        public static void CreatPoint(AxMapXLib.AxMap myMap, MapXLib.Layer
mylyr, string myKeyValue, double Mapx, double MapY)
        {
            MapXLib.PointClass pt = new MapXLib.PointClass();
            pt.Set(Mapx, MapY);

            MapXLib.Feature tempFeature = myMap.FeatureFactory.CreateSymbol(pt,
myMap.DefaultStyle);
            tempFeature.KeyValue = myKeyValue;
            mylyr.AddFeature(tempFeature, new MapXLib.RowValues());
            myMap.Update();
        }
```

附件五：

具体实现的代码如下。

```
create procedure sp_pricestate
@daterange nvarchar(1000) = null,    //计算均价的区间
@estateType nvarchar(20) = null,        //物业类型
@countrys nvarchar(2000) = null,     //区县
@regions nvarchar(2000) = null,      //区域
@ringroad nvarchar(2000) = null,     //环线
@statType bit = 1,                   //统计租金 =0、售价 =1
@idatelen int = 7                    //按年、按月（默认）
as
set nocount on

declare @place nvarchar(2000)//以逗号分隔的区县、区域、环线字符串
declare @aplace nvarchar(30)//区县、区域、环线
declare @index   int
declare @adate nvarchar(30)//计算均价的月份
declare @dindex   int
declare @avgprice int , @avghire int , @statAvage int//售价、租金、统计均价
declare @icount int
declare @daterange_copy nvarchar(1000)//计算均价期间的副本
set @daterange_copy = @daterange

if (@idatelen <> 7 and @idatelen <> 4) return
//临时表存储计算结果
create table #tmp
(
    regionmae nvarchar(20),
    avgprice1 int,
    ......
    avgprice24 int,
)

select @countrys = isnull(@countrys,''),@regions = isnull(@regions,'')
declare @ngrouptype int
if len(@countrys) > 0
    begin
        select @place = @countrys , @ngrouptype = 1 //区县
    end
else if len(@regions) > 0
    begin
        select @place = @regions , @ngrouptype = 2   //区域
    end
else if len(@ringroad) > 0
```

```
        begin
            select @place = @ringroad , @ngrouptype = 3 //环线
        end

    //遍历所有的区县
    while len(@place) > 1
    begin
        set @index = charindex(',' , @place )
        set @aplace = replace( left(@place ,   @index - 1) ,'区','')
        set @icount = 0
        print '@aplace    = ' + @aplace

        set @daterange = @daterange_copy
        //逐个月份计算均价
        while len(@daterange) > 1
        begin
            //解析计算区间字符串，获得下一个要计算的月份
            set @dindex = charindex(',' , @daterange )
            set @adate = left(@daterange ,   @dindex - 1)

            set @icount = @icount + 1
            select @avgprice = 0 , @avghire = 0

            if (@estateType = '住宅') //计算住宅的均价
            begin
    select @avgprice = sum( isnull(BuildingPrice,0) * isnull(buildingSize,0)) /
sum(cast(buildingSize as numeric(18,2)))
                    from house
                    where isnumeric(BuildingPrice) = 1
                    and isnumeric(buildingSize) = 1 and charindex('e',buildingSize)
= 0
        and (County like @aplace + '%' or SerialNo like @aplace + '%' or RingRoad
like @aplace + '%')
                    and surveyDate like @adate + '%'
                    having sum(cast(buildingSize as numeric(18,2)))   > 0
            end
            else if (@estateType = '写字楼')
            begin
                if @statType = 1
    //计算写字楼销售均价
    ......
                else
    //计算租写字楼金均价
    ......
```

```
            end
            else if (@estateType = '商业')
            begin
                if @statType = 1
    //计算商业销售均价
    ......
                Else
                    //计算商业租金均价
    ......
            end
            else if (@estateType = '工业')
            begin
                select @avgprice = 0 , @avghire = 0
            end

            select @avgprice = isnull(@avgprice , 0) , @avghire =
isnull(@avghire,0)
            //计算销售均价还是计算租金均价
            if @statType = 1
                set @statAvage = @avgprice
            else
                set @statAvage = @avghire

            //将计算结果写入临时表
            if(@icount = 1)
                insert into #tmp (regionmae , avgprice1) values (@aplace ,
@statAvage)
            else if(@icount = 2)
                update #tmp set avgprice2 = @statAvage where regionmae =
@aplace
            ......
            else if(@icount = 24)
                update #tmp set avgprice24 = @statAvage where regionmae =
@aplace

    set @daterange = substring(@daterange, @dindex + 1, len(@daterange) -
@dindex)
        end

        set @place = substring(@place, @index + 1, len(@place) - @index)
    end

    if (@ngrouptype = 2) //区域
    begin
```

```
        update #tmp
        set regionmae = substring(regionDetails,charindex(' ',regionDetails) + 1 ,
len(regionDetails) - charindex(' ',regionDetails))
        from #tmp , regionCatalog
        where regionmae = substring(regionDetails,1,charindex(' ',regionDetails) - 1)
    end
    //将临时表中销售均价的计算结果返回
    select * from #tmp order by avgprice1 asc

    set nocount off
    go
```

附件六：

```
create   procedure sp_shoujiaPoint
@nchdate nvarchar(20)   //年月
as
set nocount on
//变量定义
......
if(len(@nchdate) != 7 or charindex('.',@nchdate) =0)
begin
    select '参数不对，日期长度必须等于 7。比如[ 2008.08 ]'
    return
end

if(exists( select 1 from shoujiaAvg where avgDate like @nchdate + '%' ))
begin
    //删除可能计算过的数据
    delete from shoujiaCompare where cmpDate >= @nchdate
    delete from shoujiaAvg where avgDate >= @nchdate
end

select @avgHouse1 = 0 , @avgHouse2 = 0 ,@avgBusiness1 = 0 ,
@avgBusiness2 = 0 , @avgOffice1 = 0 , @avgOffice2 = 0

select @cmpRingHouse = 0 , @cmpSameHouse = 0 , @cmpShoujiaHouse = 0
select @cmpRingOffice = 0 , @cmpSameOffice = 0 , @cmpShoujiaOffice = 0
select @cmpRingBusiness = 0 , @cmpSameBusiness = 0 ,
@cmpShoujiaBusiness = 0

//计算去年同月
set @lastYear = cast( (cast(substring (@nchdate , 1 ,4) as int) - 1)   as nvarchar)
+ substring( @nchdate , 5,3)
//首佳指数基准月（参考月）为 2008.01
set @baseMonth = '2008.01'

//计算上个月
select @tmpYear = case cast(substring( @nchdate , 6,2) as int) when 1 then
(cast(substring (@nchdate , 1 ,4) as int) - 1) else substring (@nchdate , 1 ,4) end
select @tmpMonth = case cast(substring( @nchdate , 6,2) as int) when 1 then 12
else (cast(substring( @nchdate , 6,2) as int) - 1) end

if len(@tmpMonth) = 1 select @tmpMonth = '0' + @tmpMonth
set @lastMonth = @tmpYear + '.' +   @tmpMonth

//计算'住宅'的加权平均值
```

```
    select @avgHouse1 = sum( isnull(BuildingPrice,0) * isnull(buildingSize,0)) /
sum(cast(buildingSize as numeric(18,2)))
        from house
        where isnumeric(BuildingPrice) = 1   and   charindex('e',BuildingPrice) = 0
        and isnumeric(buildingSize) = 1 and   charindex('e',buildingSize) = 0
        and surveyDate like @nchdate + '%'
        having sum(cast(buildingSize as numeric(18,2)))   > 0
    //计算'住宅'的算数平均值
    select @avgHouse2 = sum( isnull(BuildingPrice,0 )) / count(*)
        from house
        where isnumeric(BuildingPrice) = 1   and   charindex('e',BuildingPrice) = 0
        and isnull(BuildingPrice,0) > 0
        and surveyDate like @nchdate + '%'
        having count(*) > 0

    //计算'写字楼'、'商业'的加权平均值
    ......
    //计算'写字楼'、'商业'的算数平均值
    ......
    if (@avgHouse1 <=0 and @avgHouse2 <=0 and @avgBusiness1 <=0 and
@avgBusiness2 <=0 and @avgOffice1 <=0 and @avgOffice2 <=0)
    begin
        select '平均值有 0（数据不全或有问题）。'
        return
    end

    insert into shoujiaAvg( [avgDate] , [avgType], [avgHouse], [avgBusiness],
[avgOffice], [avgPlay])
        values(@nchdate , '加权平均',@avgHouse1 , @avgBusiness1 ,
@avgOffice1 ,1)

    insert into shoujiaAvg([avgDate] ,   [avgType], [avgHouse], [avgBusiness],
[avgOffice], [avgPlay])
        values(@nchdate , '算术平均',@avgHouse2 , @avgBusiness2 ,
@avgOffice2 ,0)

    set @compareType = '加权平均'

    while len(@compareType) > 1
    begin
        set @compareObject = '住宅'
        select @currentAvg = avgHouse from shoujiaAvg where avgDate =
@nchdate and avgType = @compareType
        //计算环比指数
```

```
        set @lastAvg = 0
        select @lastAvg = avgHouse from shoujiaAvg where avgDate = @lastMonth
and avgType = @compareType
        if ( @lastAvg > 0 and @currentAvg > 0) set @cmpRingHouse =
(((@currentAvg - @lastAvg) / @lastAvg) * 100
        //计算同比指数
        set @lastAvg = 0
        select @lastAvg = avgHouse from shoujiaAvg where avgDate = @lastYear
and avgType = @compareType
        if ( @lastAvg > 0 and @currentAvg > 0) set @cmpSameHouse =
(((@currentAvg - @lastAvg) / @lastAvg) * 100
        set @lastAvg = 0
        //计算首佳指数
        set @lastAvg = 0
        select @lastAvg = avgHouse from shoujiaAvg where avgDate =
@baseMonth and avgType = @compareType
        if ( @lastAvg > 0 and @currentAvg > 0) set @cmpShoujiaHouse =
(@currentAvg / @lastAvg) * 100
        //保存指数计算结果
    insert into   [shoujiaCompare]([cmpDate], [cmpObject], [cmpType],[avgPrice],
[ cmpRing], [cmpSame], [cmpShoujia])
            values( @nchdate , @compareObject , @compareType ,@currentAvg ,
@cmpRingHouse , @cmpSameHouse , @cmpShoujiaHouse)

        //'写字楼、商业的环比、同比、首佳指数的计算
    ......
        if @compareType = '加权平均'
            set @compareType = '算术平均'
        else
            set @compareType   = ''
    end

    set nocount off
    go
```