

**本 科 毕 业 论 文**

基于Android的停车管理系统

**A Parking Management System Based on Android Operating System**

学院名称： 电子信息与电气工程学院

专业班级： 通信工程（专升本）2012级1班

学生姓名： 黄宁韬

学 号： 201202080022

指导教师姓名： 申庆超

指导教师职称： 讲师

2014年 5月

**毕业设计（论文）原创性声明和使用授权说明**

**原创性声明**

本人郑重承诺：所呈交的毕业设计（论文），是我个人在指导教师的指导下进行的研究工作及取得的成果。尽我所知，除文中特别加以标注和致谢的地方外，不包含其他人或组织已经发表或公布过的研究成果，也不包含我为获得安阳工学院及其它教育机构的学位或学历而使用过的材料。对本研究提供过帮助和做出过贡献的个人或集体，均已在文中作了明确的说明并表示了谢意。

作 者 签 名：　　 　 　　 日　 期：

指导教师签名：　　 　 　　 日　　期：

**使用授权说明**

本人完全了解安阳工学院关于收集、保存、使用毕业设计（论文）的规定，即：按照学校要求提交毕业设计（论文）的印刷本和电子版本；学校有权保存毕业设计（论文）的印刷本和电子版，并提供目录检索与阅览服务；学校可以采用影印、缩印、数字化或其它复制手段保存论文；在不以赢利为目的前提下，学校可以公布论文的部分或全部内容。

作者签名：　　 　 　　 日　 期：

**目录**

[摘要 5](#_Toc386531249)

[Abstract: 6](#_Toc386531251)

[引言 7](#_Toc386531252)

[第一章 绪论 8](#_Toc386531253)

基于Android的停车管理系统 5

A Parking Management System Based on Android Operating System 6

Abstract: 6

引言 7

第一章 绪论 8

1.1所选课题的研究背景与意义 8

1.2设计系统所要达到的要求 9

1.3 C/S结构简介 9

1.4 本文研究内容及章节安排 9

第二章 开发平台和开发语言简介 10

2.1 Android操作系统的介绍 10

2.1.1 Android简介 10

2.1.2 Android的特点 10

2.1.3 Android架构 10

2.2开发工具和开发语言的介绍 11

2.2.1 客户端软件开发工具Eclipse简介 11

2.2.2 服务端软件开发工具Visual Studio简介 12

2.2.2 开发语言简介 12

第三章 开发中的关键技术 13

3.1 客户端与服务端之间的通信技术 13

3.1.1 套接字通信原理 13

3.1.2 数据报通信 13

3.2 多线程编程 13

3.2.1 Java和C#中的Thread类 14

3.3 停车场用户数据存储 14

3.3.1 数据库SQLite 14

3.3.2 XML文件 15

第四章 停车管理系统的设计 16

4.1 系统设计概述 16

4.2 系统设计层次图 16

4.3 各模块详细功能 17

4.3.1 服务端 17

4.3.2 客户端 18

4.4 主要模块分析 18

4.4.1 服务端数据保存模块 18

4.4.2 服务端监听模块 20

4.4.3 客户端汽车进库模块 22

4.4.4 客户端接监听提醒模块 22

4.4.4 客户端用户数据保存模块 24

第五章 停车管理系统的测试运行 26

5.1 服务端运行界面 26

5.2 客户端运行界面 27

结论 29

致谢 30

参考文献 31

基于Android的停车管理系统

**摘要：**随着我国现代化进程的不断加快，私家车的数量急剧增加，各类停车场的车流量不断增多，对机动车的保管越来越成为社会倍受关注的问题。

本论文主要介绍了Android系统架构，并且针对现在普遍采用的人工收费管理方法存在的弊端，提出了一套由Android SDK和Visual C#开发设计的车库车位管理信息系统，使用Socket技术进行电脑服务端和手机客户端之间的通信，手机端软件和电脑端软件分别采用SQLite和XML保存用户数据，采用流行的C/S 结构，也就是客户端/服务器模式。该系统在入口处由用户使用智能手机输入用户的姓名，车牌号，手机号等信息，在出口处服务端根据车牌号在停车记录中自动查找该车信息，并自动计费，收取停车费，当用户停车超过一定时间时，系统会自动发送短信提醒。该系统共分为登录模块、计费模块、查询模块、用户管理模块、短信提醒模块，文中对各模块的功能、操作界面、工作流程做了详细的介绍。

**关键词：**C/S结构；Android操作系统；Socket通信；停车管理

A Parking Management System Based on Android Operating System

Abstract: With the development of economy, the number of cars increasing drastically, so a serious problem emerge: how to supervise the park of cars.

This paper mainly introduce the framework of Android Operating System and bring out a new car management system that can replace the old which is done by human power. The system is based on Android SDK and Visual C# development platform. It uses Socket technology to communicate between mobile phone and computer and employs XML file on computer, SQLite on mobile phone accordingly to store user data. The system needs user entering their name, phone number, car number when they arrive, and calculate parking time and fee when they leave. This system contains login module, counting fee module, query module, user management module and SMS notification module.

**Keywords:** C/S Architecture**;** Android Operating System; Socket Communication; Garage Management

引言

在现代社会中管理的作用越来越显得重要和突出。一般来讲，管理通过计划、组织、指导与领导、控制等手段，为组织制定目标，应用组织的各种要素，以实现组织的目标。对企业来说，人们按照一定的营销计划、人力、物力资源，对人员、物质、资金等加以计划和协调，以达到预期的目的，即称为企业管理。传统的企业管理活动中，把人、才、物作为企业的主要资源。但是随着社会化大生产的不断扩大和社会对产品多样化的需求，人们越来越重视信息在生产经营及企业管理中的作用，并把它当做企业的一种极其重要的资源，人们称之为“信息资源”，信息资源的处理已经成为当今世界上一项主要的社会活动。同样，停车场中也不例外，其中加强对车位信息的管理，一方面能更好为车主服务，另一方面能加强对停车场效益的监控。

本世纪以来，由于社会生产力的迅速发展和科学技术的突飞猛进，人们进行信息交流的深度和广度不断增加，管理所需要的信息量急剧增长，同时对信息的处理要求及时、准确，这导致了传统的信息处理方法和手段已不能适应现代管理的需要；电子计算机的诞生和在信息处理中的应用，标志着一个崭新时代——“信息时代”的开始，它使企业管理现代化成为可能。

一定意义上说，管理信息系统的产生和发展是建立在电子计算机基础之上的。随着智能手机一步步走进普通人的生活，它将成为人们获取信息的主要设备。因此，手机的娱乐应用服务将会有很大的发展空间，基于智能手机的停车管理软件就是其中之一

# 绪论

## 1.1所选课题的研究背景与意义

信息作为生产力中最活跃的因素，很早就在人类经济生活中发挥着不可替代的作用，但把信息纳入企业经营管理系统中高效处理和应用是在计算机发明以后的50年代末。

目前人们所提到的管理信息系统，通常是指以计算机为工具，对管理信息进行收集、存贮、检索、加工和传递，使其应用于组织机构及企业管理领域的“人－机系统”，该系统是70年代初“后工业经济”时代的产物，在企业管理信息系统在其发展的初期，是单一的人—机系统，系统功能简单，效率低，对管理的作用有限。

进入20世纪80年代末，随着计算机网络和通信技术的大力发展，企业管理信息系统进入网络化的发展时期。世界发达国家已建立起了完善的先进的管理信息系统体系，在其社会经济发展领域中发挥着巨大的作用。

我国的管理信息系统大发展是在80年代末90年代初，国家相继建立起了“金卡”、“金关”、金桥“工程，多数企事业单位建立了局域网和广域网管理信息系统。随着世界经济和我国经济逐步向知识经济迈进，无论是基于工业经济的管理信息系统，还是面向少数专家和管理人员的专家系统 (ES)或决策支持系统 (DSS)，其处理对象和服务对象，自身的系统结构，处理能力，都有了进一步的发展。

停车场管理（Car Park Management）并不是一个新鲜词。想想看，您是如何管理您的车辆信息的？其实您在日常生活中不自觉地运用着各种管理方法。但科学管理的前提是建立正确的管理观、正确管理科学管理的含义。

如何科学管理？

正确的管理观念非常重要，科学管理方式选择将成为决定车辆贫富差距的关键性因素。

停车场管理的核心是投资收益的最大化和车辆信息分配合理化的集合。管理服务通过充分利用各种管理工具，帮助您达到合理分配的目的、满足您对管理安全性、收益性等多样化要求。

开发一个停车场车辆管理系统迎合了现代的公司发展趋势，对居民合理管理公司和他人的联系信息起到重要意义。

## 1.2设计系统所要达到的要求

目标系统应该达到以下要求：

1、时间经济性。优化逻辑设计与物理设计，使系统运行效率高，反映速度快。 2、可靠性。能连续准确的处理业务，有较强的容错能力。 3、可理解性。用户容易理解和使用该系统。

4、可维护性和适应性。系统应易于修改、易于扩充、易于维护，能够适应业务不断发展变化的需要。

5、可用性。目标系统功能齐全，能够完全满足业务需求。

6、安全保密性。保证系统的物理安全、数据存储和存取的安全与保密、数据传输的安全与保密，做好使用人员的授权管理。

## 1.3 C/S结构简介

C/S 结构，即众所周知的客户机和服务器结构。它是软件系统体系结构，通过它可以充分利用两端硬件环境的优势，将任务合理分配到Client端和Server端来实现，降低了系统的通讯开销。目前大多数应用软件系统都是Client/Server形式的两层结构，由于现在的软件应用系统正在向分布式的Web应用发展，Web和Client/Server 应用都可以进行同样的业务处理，应用不同的模块共享逻辑组件；因此，内部的和外部的用户都可以访问新的和现有的应用系统，通过现有应用系统中的逻辑可以扩展出新的应用系统。这也就是目前应用系统的发展方向。

## 1.4 本文研究内容及章节安排

本文所实现的是基于智能手机平台的停车管理系统，并对当前智能手机软件开发环境进行了研究。

第一章是绪论，首先论述本课题的研究背景以及意义，其次介绍了本系统所达到的设计要求，最后给出了本文的结构和主要内容安排。

# 开发平台和开发语言简介

## 2.1 Android操作系统的介绍

### 2.1.1 Android简介

Android是Google在2007年11月宣布的基于Linux平台的开源手机操作系统的名称，而在Android系统底层方面，Android使用C/C++作为开发语言。该平台由操作系统、中间件、用户界面和应用软件组成，号称是首个为移动终端打造的真正开放和完整的解决方案。

Android系统就是一个开发式的手机和平台电脑操作系统[5]，早期由Google开发，后由开放手机联盟开发。它提供了从底层操作系统到上层的程序界面的所有软件，使用这个平台无需缴纳任何的授权许可费用，同时不同的厂商还可以根据自身的需求修改和扩展Android平台。

### 2.1.2 Android的特点

Android开发平台是由谷歌与开放手机联盟合作开发的一个开放、自由的移动终端平台，它由操作系统、中间件、应用软件三部分组成。该平台备有完善的程序开发环境，包括模拟器、除错工具、程序执行所耗用存储器、性能评介等功能。Android为开发人员提供了大量的使用库和工具，使得开发人员可以快速地创建自己的应用程序。

在Android平台下，除了应用程序运行的载体虚拟机之外，所有的Android应用程序之间是完全平等的。所有的应用程序都运行在虚拟机上面，虚拟机提供了一系列用于应用程序和硬件资源间通信的API。Android打破了应用程序之间的界限，在Android平台下开发应用程序，可以很方便的实现应用程序之间的数据共享。一个应用程序不但可以通过API访问系统提供的功能，还可以通过申明自身的功能供其他应用程序调用

### 2.1.3 Android架构

Android 是首个基于 Linux 平台的开源手机操作系统，该平台由操作系统、中间件、用户界面和应用软件组成，是首个为移动终端打造的真正开放和完整的软件。Android手机系统是基于Linux平台的开源手机操作系统,分为4个层,下面是对于每一层的具体介绍：

第一层：底层的Linux kernel，采用的是Linux kernel 2.6版，也是目前最新的版本， 主要的功能包括硬件的驱动、网络、电源、以及系统安全和内存管理等。 第二层：Library函数库，有大多数幵放源代码的函数库所组成，例如标准的C函 数库Libc、OpenSSL、SQLite等，还包括WebKit函数库，负责Android网页浏览器的 运行，还包括有OpenGL图形库等。当然在这层还有很重要的Android Runtime，在此提 供Android特有的Java内核函数库与可转换Java运行码Bytecode，为Android专属的 Dalvik Executable (.dex)文件格式的Dalvik虚拟机，可以通过Java语言编写Android 应用程序。

第三层则是Application Framework (应用软件架构），它是所有Android重点应用程 序Framework API的集合，方便开发者使用这些API进行设计架构。常见的API包括了 显示功能（按钮、内嵌式浏览器），消息提示功能（访问信息、分享信息），提示信息（显示警告信息），位置管理等。

第四层：Java应用程序，例如Google Maps, E-mail,浏览器、等应用程序。

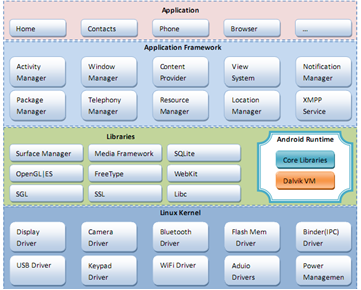
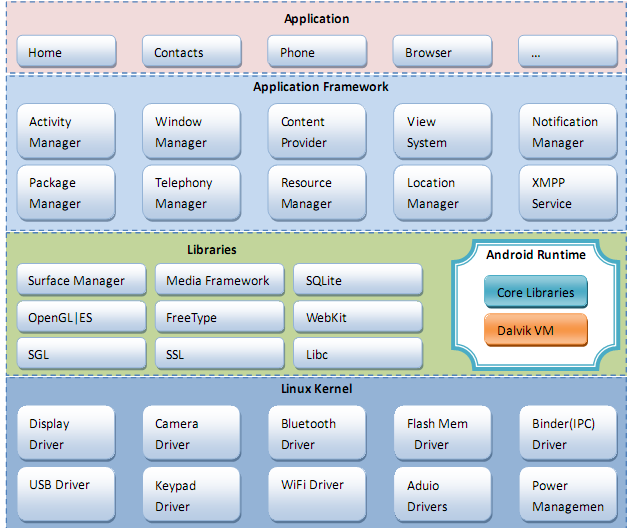


图2.1 Android系统层次结构

## 2.2开发工具和开发语言的介绍

### 2.2.1 客户端软件开发工具Eclipse简介

Eclipse 最初是IBM的一个软件产品，前期投入了4000万美金。2001年11月，IBM宣布将其捐给开放源码组织Eclipse.org。目前Eclipse的市场占有率已经超过了Borland公司的JBuilder，成为Java开发平台中的主流。Eclipse的设计思想是：一切皆为插件。它自身的核心是非常小的，其它所有的功能都以插件的形式附加到该核心上。Eclipse是一个开放源代码的、基于Java的可扩展开发平台。就其本身而言，它只是一个框架和一组服务，用于通过插件组件构建开发环境。幸运的是，Eclipse附带了一个标准的插件集，包括Java开发工具（Java DevelopmentTools，JDT）。虽然大多数用户很乐于将Eclipse当作Java IDE来使用，但Eclipse的目标不仅限于此。Eclipse还包括插件开发环境（Plug-in DevelopmentEnvironment，PDE），这个组件主要针对希望扩展Eclipse的软件开发人员，因为Android手机查询公交信息系统的设计与实现6它允许他们构建与Eclipse环境无缝集成的工具。由于Eclipse中的每样东西都是插件，对于给Eclipse提供插件，以及给用户提供一致和统一的集成开发环境而言所有工具开发人员都具有同等的发挥场所。这种平等和一致性并不仅限于Java开发工具。Eclipse框架还可用来作为与软件开发无关的其他应用程序类型的基础。Android开发平台提供了在Eclipse中的集成工具ADT(Android DevelopmentTools)，在Eclipse中开发Android应用程序需要嵌入ADT。

本文选择Eclipse作为开发工具，主要原因是Edipse将会是未来最流行的集成开发工具之一。它所具有的漂亮的界面、免费且强大的功能、国际化、灵活的插件式体系结构、支持多种操作系统平台等特点，无一不是其魅力所在。

### 2.2.2 服务端软件开发工具Visual Studio简介

Microsoft Visual Studio（简称VS）是美国微软公司的开发工具包系列产品。VS是一个基本完整的开发工具集，它包括了整个软件生命周期中所需要的大部分工具，如UML工具、代码管控工具、集成开发环境(IDE)等等。所写的目标代码适用于微软支持的所有平台，包括Microsoft Windows、Windows Mobile、Windows CE、.NET Framework、.NET Compact Framework和Microsoft Silverlight 及Windows Phone。

### 2.2.2 开发语言简介

手机端软件以JAVA为基础的，但是开发环境不一样。在编写的过程中不是单纯的Java语言，而是与android相结合。

电脑端软件使用c#语言开发Windows窗体应用程序

# 开发中的关键技术

## 3.1 客户端与服务端之间的通信技术

### 3.1.1 套接字通信原理

套接字的英文名为“socket”。套接字允许程序员把网络连接当成一个流，并向这个流读写字节。

  我们都知道在Java中有两个重要的类Socket和ServerSocket。Socket和ServerSocket都被放在java.net包中。通过这两个类就能完成客户端与服务器、甚至是客户端与客户端的即时通讯。首先通过ServerSocket类在服务器某固定端口等待连接，当有Socket请求连接到此服务器的此端口时，就完成Socket连接，然后相互之间就可以发送和接收信息。

Socket类是Java用来实现客户端TCP操作的基础类，在Java编程中，一切与TCP有关的操作，包括连接的建立、传输、连接关闭及Socket选项的设置都由Socket类负责，这个类本身使用直接代码通过主机操作系统的本地TCP栈进行通信。Socket类的消息收发都是通过流来实现。通过new Socket()创建需要连接的指定的服务器host和端口号port的对象，并向服务器发送连接请求。通过close()方法关闭已建立的套接字。

ServerSocket类能够在服务器端指定的端口监听。ServerSocket类通过new ServerSocke()能实现等待某客户端的连接请求的功能。跟Socket一样，也是通过close()方法关闭。

在C#中，与Socket通信相关的函数使用方法与Java中完全相同。

### 3.1.2 数据报通信

在TCP/IP协议中有一个重要的定义，那就是IP数据报(IP Datagram)。IP数据报是在Internet上传输的包。它和硬件的配置无关，是由两部分组成：首部和数据。首部的前20个字节是固定的，它是所有IP数据报都必须要有的，首部剩下部分的长度是可变的。根据需要进行相应分配。例如考虑排错或者安全等因素需要。

## 3.2 多线程编程

多线程的目的是使对CPU的利用率提高到最大。操作系统由原来的单任务发展到现在能够同时进行多任务。实现多任务就可以通过多线程。我们知道进程是指在电脑内存运行的一个应用程序。在内存中每个进程都是独立占用的内存空间。一个进程可以有很多个线程在运行。例如火车的售票系统，在售票系统运行过程中，在同一时间可能有很多人在进行买票操作。通过多线程方式能很好解决同时进行的操作。在操作系统中，每个线程都会有一个优先级。优先级高的线程在正常情况下比优先级低的先执行。

Java实现多线程有两种方式：通过继承java.lang.Thread类和实现java.lang.Runnable接口。而C#主要使用System.Threading类的.Start()方法运行线程

### 3.2.1 Java和C#中的Thread类

Thread类提供了用来启动、挂起、恢复以及终止线程的方法，除此之外，还提供控制线程的优先级以及线程的名字等其他方面的方法。使用Thread类最简单的方法是扩展它，并覆盖run()方法，当线程启动时会调用这个方法。通过覆盖run()方法，就可以使线程在后台完成一些有用的任务。而C#则需要在Thread类的构造方法里指定一个运行线程所需要的目标函数，如Thread listenerThread = new Thread(new ThreadStart(listener));其中listener即为目标函数。

## 3.3 停车场用户数据存储

### 3.3.1 数据库SQLite

Android 数据库使用的是系统内嵌的 SQLite，下面简单的介绍 Android 平台下的 SQLite 数据库的特点。

SQLite是遵守的关系型数据库管理系统，它包含在一个相对小的C库中。不像常见的客户-服务器范例，SQLite引擎不是个程序与之通信的独立进程，而是连接到程序中成为它的一个主要部分。SQLite 是一个非常流行的嵌入式数据库，它支持 SQL 语言，并且只利用很少的内存就有很好的性能。此外它还是开源的，任何人都可以使用它。SQLite 由以下几个组件组成：SQL 编译器、内核、后端以及附件。SQLite 通过利用虚拟机和虚拟数据库引擎（VDBE），使调试、修改和扩展 SQLite 的内核变得更加方便。

Android 运行时环境包含了完整的 SQLite。Android 在运行时（Run-Time）集成了 SQLite，所以每个 Android 应用程序都可以使用 SQLite 数据库。对于熟悉 SQL 的开发人员来时，在 Android 开发中使用 SQLite 相当简单。这样的环境更有利于开发人员对项目的开发。

### 3.3.2 XML文件

1998年2月，W3C正式批准了可扩展标记语言的标准定义，可扩展标记语言可以对文档和数据进行结构化处理，从而能够在部门、客户和供应商之间进行交换，实现动态内容生成，企业集成和应用开发。可扩展标记语言可以使我们能够更准确的搜索，更方便的传送软件组件，更好的描述一些事物。例如电子商务交易等。它被设计用来传输和存储数据；

超文本标记语言被设计用来显示数据。它们都是标准通用标记语言的子集。

# 停车管理系统的设计

## 4.1 系统设计概述

智能停车管理系统的设计跟普通程序的设计有着许多不同的地方，采用C/S结构的智能停车管理需要考虑到手机数据与电脑数据之间共享的问题，当用户拿着智能手机在入口处登记用户信息时，需要将手机上的数据上传到服务端中，当汽车出库时，手机要监听服务端发送过来的出库信息，才能够将出库信息即使提醒用户。因此，在设计停车管理系统的过程中，必须要考虑好以下几个设计要点：

在网络中的管理系统程序需要以服务器提供服务端连接响应，使用者通过客户端程序与服务器建立起联系，这样才能实现手机与电脑之间数据的共享，这是一个面向连接的通信过程。因此，程序要在TCP/IP环境下，实现服务器端和客户端两部分程序。

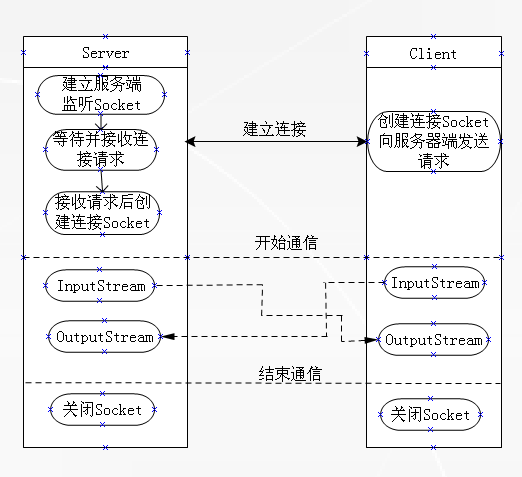


图4.1 套接字通信模型

## 4.2 系统设计层次图

停车管理系统

客户端

服务器端

用户数据保存

监听服务端进出库情况

汽车出库请求

入库数据输入

短信提醒

用户出库

停车计费

用户信息显示

用户数据保存

图4.2 系统总体模型

## 4.3 各模块详细功能

### 4.3.1 服务端

服务器监听控制窗体

ServerFrame

启动服务器监听

AppServer

处理所有用户请求

Connection

图4.3 服务端系统模型

服务端程序清单：

Program.cs：服务端主界面，负责用户进库，出库及计费。

SocketHost.cs：服务器端监听客户端连接，并处理各种请求。

SocketResponse.cs：服务器端发送客户端消息。

XmlParser.cs: XML解析程序

### 4.3.2 客户端

启动监听服务

StartService

入库模块

LoginActivity

用户出库模块

LogoutActivity

用户主界面

HomeActivity

客户端监听模块

SocketServer

图4.4 客户端系统模型

客户端程序清单：

HomeActivity.java：客户端主界面

LoginActivity.java：用户进库主界面

LogoutActivity.java：用户出库主界面

SocketServer.java：Serivce服务，负责监听服务端发送的出库信息

AppUtil.java：工具集程序

SendSMSTask.java：发送短信提醒的定时器

SqlHelper.java：数据库操作类

## 4.4 主要模块分析

### 4.4.1 服务端数据保存模块

经分析发现每个用户的进库信息只有姓名，车牌号码，手机号，和进库时间这四个项目，数据量并不大，所以服务端用户采用轻量级XML文件存储用户数据的方法，这样做的好处是服务端不需要部署数据库，降低了使用成本。

C#使用XmlDocument类对XML文件进行解析。XmlDocument将XML视为树状结构，它装载XML文档，并在内存中构建该文档的树状结构。

在本程序中，我们首先检测程序路径下是否存在data.xml文件，如果不存在则先在xml容器中创建XmlDeclaration并且调用CreateElement("", "GarageInfo", "");创建名为GarageInfo的根节点，之后存入data.xml中。一个新创建的data.xml如下：

<?xml version="1.0" encoding="UTF-8"?>

<GarageInfo />

每次启动程序是，我们需要对data.xml文件进行解析，我们使用方法User [] parse\_xml(filename)，该方法返回一个用户数据结构数组。通过该数组能够获得xml文件中所有用户信息。parse\_xml()方法的代码如下：

private User[] parse\_xml(String filename) {

XmlDocument doc = new XmlDocument();

doc.Load(filename);

XmlNode node = doc.SelectSingleNode("/GarageInfo");

XmlNodeList nodeList = node.ChildNodes;

int i = nodeList.Count;

User[] all\_usr = new User[i];

//read all data into User[] array

for(int j=0;j<i;j++)

{

XmlNode internal\_node = nodeList.Item(j);

if (internal\_node != null)

{

User user = new User();

user.NAME = internal\_node.Attributes["name"].Value;

user.CAR\_NUMBER = internal\_node.Attributes["carnum"].Value;

user.PHONE\_NUMBER = internal\_node.Attributes["phonenum"].Value;

user.TIME = internal\_node.Attributes["TIME"].Value;

all\_usr[j] = user;

}

}

return all\_usr;

}

当客户端发送用户进库信息时，我们要把用户进库信息写入xml文件，我们使用write\_xml(String NAME, String CAR\_NUMBER, String PHONE\_NUMBER, String TIME)方法。该方法调用需要提供用户名，车牌号，电话号码，以及入库时间。该方法代码如下：

private void write\_xml(String NAME, String CAR\_NUMBER, String PHONE\_NUMBER, String TIME)

{

//加入另外一个元素

XmlDocument doc = new XmlDocument();

doc.Load("data.xml");

XmlNode root = doc.SelectSingleNode("GarageInfo");

XmlElement xe1 = doc.CreateElement("User");//创建一个<User>节点

xe1.SetAttribute("name", NAME);

xe1.SetAttribute("carnum", CAR\_NUMBER); xe1.SetAttribute("phonenum", PHONE\_NUMBER);

xe1.SetAttribute("TIME", TIME);

root.AppendChild(xe1);

doc.Save("data.xml");

}

### 4.4.2 服务端监听模块

在用户点击服务端上的侦听按钮之后，服务端会初始化一个ServerSocket，并开始对端口7631进行监听，为了使监听程序不至于阻塞主线程，我们需要新开一个线程，在此线程中将监听程序放入一个while(true)循环中。一旦遇到有客户端与服务端的TCP三次握手连接完成时，serverSocket.Accept()便会被触发，并返回一个客户端clientSocket，当每建立一个TCP连接时，都会新开一个线程用以接受数据。这样做可以提升服务器的并发访问量。关键代码如下：

//set connection

ipep = new IPEndPoint(IPAddress.Any, 7631);

serverSocket = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);

serverSocket.Bind(ipep);

serverSocket.Listen(10);

//start a listener thread in order to avoid jamming the main thread

listenerThread.Start();

侦听线程代码：

private void listener()

{

while (true)

{

try

{

//create a new thread when a tcp connection accepted to recive data

clientSocket = serverSocket.Accept();

Thread clientThread = new Thread(new ThreadStart(ReceiveData));

clientThread.IsBackground = true;

clientThread.Start();

}

catch (Exception ex)

{

MessageBox.Show("listening Error: " + ex.Message);

}

}

}

当我们获取了一个clientSocket之后，我们需要获得其中汽车进库的数据，我们从输入流中提取客户端所发来的信息，然后将其写入用户列表listview和xml文件中，这里我们使用了C#自带的线程同步机制Invoke()来确保多线程的安全写入。在读取数据完成后，我们关闭套接字和线程，以便释放资源。代码如下：

private void ReceiveData()

{

bool keepalive = true;

Socket s = clientSocket;

Byte[] buffer = new Byte[1024];

clientep = (IPEndPoint)s.RemoteEndPoint;

while (keepalive)

{

int bufLen = 0;

     //client socket recive

                    bufLen = s.Available;

                    s.Receive(buffer, 0, bufLen, SocketFlags.None); clientep = (IPEndPoint)s.RemoteEndPoint;

//get stuff from buffer

                string clientcommand = System.Text.Encoding.UTF8.GetString(buffer).Substring(0, bufLen);

string[] stringSeparators = new string[] { "EOF" };

string[] split = clientcommand.Split(stringSeparators, StringSplitOptions.RemoveEmptyEntries);

Invoke(new MethodInvoker(delegate()

{

add\_listitem(split[0], split[1], split[2], split[3]);

write\_xml(split[0], split[1], split[2], split[3].Trim());

}));

keepalive = false;

}

s.Shutdown(SocketShutdown.Receive);

s.Close();

Thread.CurrentThread.Abort();

}

### 4.4.3 客户端汽车进库模块

当用户在智能手机上输入用户信息进行进库操作时，我们首先从控件中提取文本信息，然后新开一个线程，将用户信息保存到数据库中，并且通过套接字将用户信息发往服务器，关键代码如下：

Runnable doBackGroundOperation = **new** Runnable() {

**public** **void** run() {

**if**(helper.insert(name, carNumber, phoneNumber, time)){

mHandler.post(**new** Runnable(){

startActivity(intent);

**try** {

// 实例化Socket

Socket socket = **new** Socket(*HOST*, *PORT*);

PrintWriter out = **new** PrintWriter(**new** BufferedWriter(

**new** OutputStreamWriter(socket.getOutputStream())), **true**);

out.println(name+"EOF"+carNumber+"EOF"+phoneNumber+"EOF"+time);

out.close();

socket.shutdownOutput();

socket.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}**else**{

}

}

Thread thread = **new** Thread(doBackGroundOperation,"BackGroundOperation");

thread.start();

### 4.4.4 客户端接监听提醒模块

在实现客户端我们利用一个Android特有的机制Service来实现对服务端消息的长期监听。一个Service也是一种应用程序组件，它运行在后台以提供某种服务，通常不具有可见的用户界面。其它的应用程序组件可以启动一个Service，即使在用户切换到另外一个应用程序后，这个Service还是一直会在后台运行。此外，一个应用程序也可以绑定到一个Service然后使用进程间通信（IPC）方式与Service之间发生交互。例如一个Service可以处理网络事物，播放音乐，读写文件或者读写ContentProvider，所以这些都在后台运行。

在本系统中，我们在HomeActivity的onCreat()方法中显式的调用startService(intent)来启动一个Service服务。在Service主程序内，我们使用与C#服务端监听模块相类似的Socket监听方法，进行数据监听。首先我们需要初始化一个ServerSocket实例，并指定监听端口24358。为了使监听程序不至于阻塞主线程，我们需要新开一个线程，在此线程中将监听程序放入一个while(true)循环中。每当遇到TCP连接请求时，serverSocket.Accept()便会被触发，在serverSocket.Accept()调用后，服务端和客户端便会完成TCP三次握手连接，并返回一个客户端clientSocket，当每建立一个TCP连接时，都会新开一个线程用以接受数据。

当数据接收完成时，Service会调用NotificationBuilder通知用户汽车出库信息，并且将数据库中的用户信息删除。关键代码如下：

**public** **class** AcceptRunnable **implements** Runnable {

**public** **void** run() {

**while** (**true**) {

**try** {

**final** Socket client = serverSocket.accept();

workHandler.post(**new** Runnable() {

**public** **void** run() {

clientSocket = client;

**new** Thread(**new** RecieverRunnable()).start();

}

});

} **catch** (IOException e) {

e.printStackTrace();

}

}

数据接收线程主要代码：

**public** **class** RecieverRunnable **implements** Runnable {

**public** **void** run() {

**while** (runnable) {

**try** {

read = **new** BufferedReader(**new** InputStreamReader(clientSocket.getInputStream()));

String inputString = read.readLine();

runnable=**false**;

} **catch** (IOException e) {

runnable = **false**;

e.printStackTrace();

}

}

### 4.4.4 客户端用户数据保存模块

客户端用户数据存储采用Android系统自带的关系型数据库SQLite，当用户初次运行程序时，需要创建数据库，当用户进出库时，我们需要对数据库进行增加，删除和查询。

经过分析，我们需要一张表，表中含有4个字段NAME，CAR\_NUMBER，PHONE\_NUMBER，TIME。分别为姓名，车牌号，电话号码，入库时间。我们以车牌号码为主键，当用户出库时需要输入车牌号码以便对数据库进行查询。

表4-1 用户数据表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 是否主键 | 是否为空 | 字段含义 |
| NAME | Varchar(50) | 否 | 否 | 车主姓名 |
| CAR\_NUMBER | varchar(50) | 是 | 是 | 车牌号 |
| PHONE\_NUMBER | varchar2(20) | 否 | 是 | 站经度 |
| TIME | Varchar3(20) | 否 | 否 | 入库时间 |

本程序的数据库操作模块在SqlHelper.java文件中。在该文件中，我们实现了对数据库进行创建，增加，删除和查询的功能。数据库关键代码如下：

数据库创建代码：

**private** **static** **final** String *DATABASE\_CREATE* = "create table " +

*DATABASE\_TABLE* + " (" + *CAR\_NUMBER* +

" TEXT primary key, " +

*NAME* + " TEXT not null, " +

*PHONE\_NUMBER* + " TEXT, " +

*TIME* + " TEXT);";

**public** SqlHelper(Context context, **int** version) {

**super**(context, *DATABASE\_NAME*, **null**, *DATABASE\_VERSION*);

}

// Called when no database exists in disk and the helper class needs

// to create a new one.

@Override

**public** **void** onCreate(SQLiteDatabase db) {

db.execSQL(*DATABASE\_CREATE*);

}

数据库查询代码：

**public** Cursor query(String carNum) **throws** SQLException {

StringBuffer bfr=**new** StringBuffer(carNum);

bfr.insert(0, "'"); bfr.append("'");

SQLiteDatabase db = getWritableDatabase();

Cursor cursor = db.rawQuery("select \* " +

"from GarageMgr where CAR\_NUMBER=" + bfr, **null**);

**if**(cursor != **null**) {

cursor.moveToFirst();

}

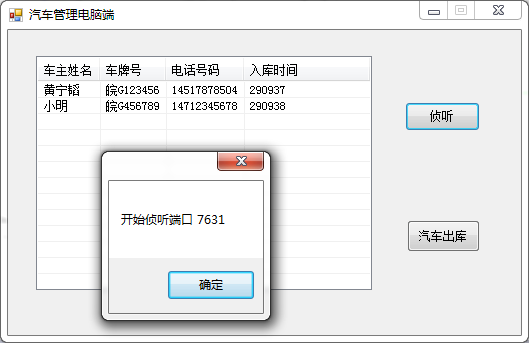
**return** cursor;

}

# 停车管理系统的测试运行

## 5.1 服务端运行界面

运行软件后直接进入主界面，主界面显示车库内的停车情况。界面右上角是监听按钮，当用户点击后即可开启服务端监听服务。界面右下角是汽车出库按钮，当用户选中一个左边列表中的用户后，用户点击出库按钮该选中用户即可出库，并告知客户端。如图5.1



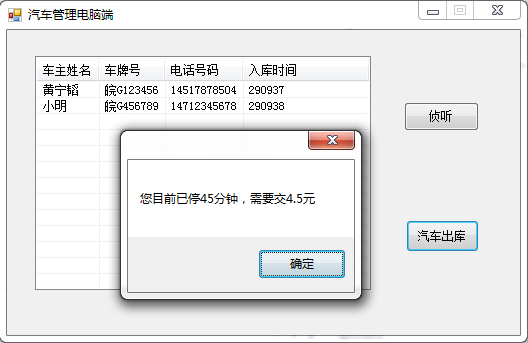
 图5.1 服务端主界面

图5.2 服务端汽车出库

## 5.2 客户端运行界面

客户端共分为三个运行界面，分别为运行主界面，汽车进库界面，汽车出库界面。在主界面中，用户可以通过点击按钮跳转进入汽车进库或者出库界面。如图6.2所示。

汽车进库界面中，用户输入自己的姓名，车牌号，手机号用来登记自己的入库信息，点击入库按钮后用户信息登记成功，系统跳转回主界面。如图6.3所示。

汽车出库界面中，用户输入自己的车牌号对数据库进行查询，当系统找到用户输入的车牌号后，系统进行出库操作并显示费用，当系统未找到该车牌号时，系统提示查询错误。如图6.4所示。

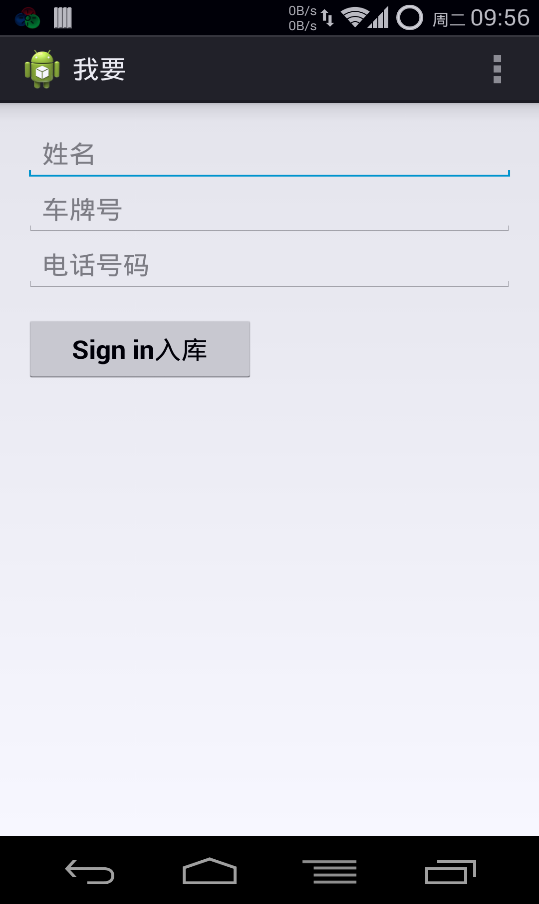
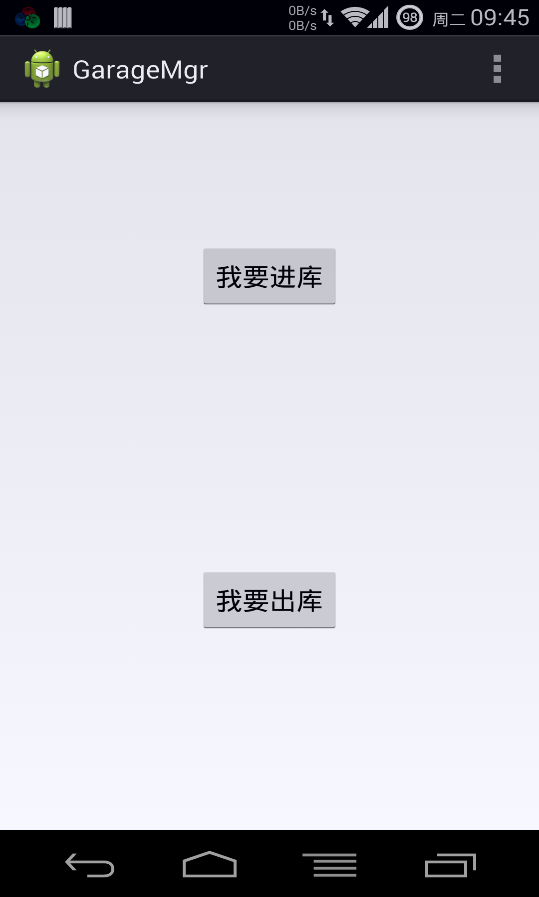


图5.3 用户主界面 图5.4 汽车入库界面



图6.4 汽车出库界面

# 结论

经过一个多月的设计和开发，系统基本开发完毕。其功能基本符合用户需求，能够完成停车场车辆管理存储和。并提供部分系统维护功能,使用户方便进行数据备份和恢复、数据删除。，对于数据的一致性的问题也通过程序进行了有效的解决。

但是由于毕业设计时间较短，所以该系统还有许多不尽如人意的地方，比如功能较少等多方面问题。这些都有待进一步改善。

在软件的编写过程中，我充分体会了软件编程工作者的辛苦，因为每一个细微的细节都必须十分的注意，如果稍有不慎，就会全盘皆输，而且往往很多的时候，程序我自己觉得非常的正确，但是就是编译通不过，在查找错误的过程中，面临着否认自己的过程，非常的痛苦，而且由于自己的经验及各方面的能力的不足，所以进展的速度非常的缓慢，往往几天的时间还没有一点进展。

在开始编写程序的时候，我看到别人的软件功能非常的详细，而且界面非常的漂亮，总希望自己的软件也非常的完善，但是，经过二个月的学习，发现编一个优秀的软件决不是一蹴而就的事情，需要长时间的积累和经验，而这个方面又是我的弱项，因此，在朋友的帮助下，我终于认清自己的不足及劣势，不要贪图大而全，只要尽我所能，编制一个实用简单的软件也是件很不容易的事情。

在反反复复的学习及虚心请教之下，我的辛勤努力有了汇报，终于做出了一个简单的软件，虽然这个软件的功能非常的简单，而且我想，在实际的运用中，肯定会有所不足，但是，它毕竟代表了我辛勤的结晶。

# 致谢

历时将近两个月的时间终于将这篇论文写完，在论文的写作过程中遇到了无数的困难和障碍，都在同学和老师的帮助下度过了。尤其要强烈感谢我的论文指导老师—申老师，他对我进行了无私的指导和帮助，不厌其烦的帮助进行论文的修改和改进。在此向帮助和指导过我的各位老师表示最中心的感谢！  
 感谢我的同学和朋友，在我写论文的过程中给予我了很多你问素材，还在论文的撰写和排版灯过程中提供热情的帮助。  
 由于我的水平有限，所写论文难免有不足之处，恳请各位老师和同学批评和指正！

参考文献

1. 孟凡荣．数据库原理与应用．中国矿业大学，2009（8）28-30
2. 盖所林．Google Android 开发入门指南 —北京：人民邮电出版社 2009.
3. 罗军舟等． TCP/IP协议及网络编程技术．清华大学出版社．2008（10）
4. 孙一林，彭波．Java网络编程实例．清华大学出版社．2003（4）
5. 张运芳．Android 创赢路线与产品开发实战[M]. 北京：电子工业出版社, 2010．
6. 李树青．Java通用范例开发金典．电子工业出版社．2009
7. 王以忠,迟成,陈绍慧,张平,李达,黄国全.  [SQLite数据库在嵌入式果蔬保鲜运输车监测系统中的应用](http://www.cnki.net/kcms/detail/detail.aspx?filename=BXJG201104004&dbcode=CJFQ&dbname=CJFD2011)[J]. 保鲜与加工. 2011(04)
8. Hofmann C, Weindorf M, Wiesel J. [Integration of GIS as a Component in Federated Information Systems](http://211.151.93.38/index.html?sid=International%20Archives%20of%20Photogrammetry%20and%20Remote%20Sensing&Title=Integration%20of%20GIS%20as%20a%20Component%20in%20Federated%20Information%20Systems&aufirst=Hofmann%20C,%20Weindorf%20M,%20Wiesel%20J&volume=2000,33&issue=9) .International Archives of Photogrammetry and Remote Sensing, 2000,33, 33 :1173~1180 .
9. Michael A P, Jeremy E W, Rocco Z. [Integration of the Global Positioning System and Geographical Information Systems for Traffic Congestion Studies](http://211.151.93.38/index.html?sid=Transportation%20Research&Title=Integration%20of%20the%20Global%20Positioning%20System%20and%20Geographical%20Information%20Systems%20for%20Traffic%20Congestion%20Studies&aufirst=Michael%20A%20P,%20Jeremy%20E%20W,%20Rocco%20Z&issue=1) .Transportation Research. 2000,86, 8(6):257~285 .
10. Lan F Akyildiz, Janise McNair, Joseph Ho, Huseyin Uzunalioglu, Wenye Wang. [Mobility Management in Current and Future Communications Networks](http://211.151.93.38/index.html?sid=IEEE%20Network&Title=Mobility%20Management%20in%20Current%20and%20Future%20Communications%20Networks&aufirst=Lan%20F%20Akyildiz,%20Janise%20McNair,%20Joseph%20Ho,%20Huseyin%20Uzunalioglu,%20Wenye%20Wang&volume=1998,124&issue=3) .IEEE Network, 1998,124, 12(4) :39~49 .
11. Levinsohn A. [Geospatial interoperability: The holy grail of GIS](http://211.151.93.38/index.html?sid=GEO%20World&Title=Geospatial%20interoperability:%20The%20holy%20grail%20of%20GIS&aufirst=Levinsohn%20A&volume=2000&issue=7) .GeoWorld, Oct, 2000