中 国 矿 业 大 学

本科生毕业设计

姓 名： **周富强** 学 号**： 08103457**

学 院： **计算机科学与技术学院**

专 业： **计算机科学与技术**

论文题目： **基于Android平台的点餐系统的设计与实现**

专 题：

指导教师： **刘兵** 职 称： **副教授**

**2014** 年 6 月 徐州

中国矿业大学毕业设计（论文）任务书

中国矿业大学毕业论文任务书

学院 计算机 专业年级 计科10-6 学生姓名 周富强

**任务下达日期：2014 年 1 月 12 日**

**毕业论文日期： 2014年 1月 18日 至 2014年 6月 2日**

**毕业论文题目：基于Android平台的点餐系统的设计与实现**

**毕业论文专题题目：**

**毕业论文主要内容和要求：**

在这样一个设计中，本文主要讲述了该点餐系统的一些相关性基础功能，以及后台数据库的使用和设计方法。

在这篇文章中，主要介绍了这个点餐系统的设计方法以及基础的功能模块。同时还介绍了这个点菜系统的数据库组织形式。这个系统最终的成型模块有具体如下的几个方面。首先就客户端来讲，有登陆模块，用来验证服务员的身份信息；点菜功能模块，用来向顾客展示菜品以及点菜；查桌，开桌和换桌功能模块，这三个模块主要是为了方便客户进行桌子的选取以及合并；还有更新模块，用来更新相关的信息；以及结账等功能性模块。

而就服务器端而言，首先只有管理员能够将登录。登陆后可以删除和添加菜品，管理服务员的信息和增删，并可以查看一些相关信息。这个是在电脑上通过网页进行操作的。

院长签字： 指导教师签字：

年 月 日

**中国矿业大学毕业设计（论文）指导教师评阅书**

**指导教师评语**（①基础理论及基本技能的掌握；②独立解决实际问题的能力；③研究内容的理论依据和技术方法；④取得的主要成果及创新点；⑤工作态度及工作量；⑥总体评价及建议成绩；⑦存在问题；⑧是否同意答辩等）：

成绩： 指导教师签字：

年 月 日

**中国矿业大学毕业设计（论文）评阅教师评阅书**

**评阅教师评语**（①选题的意义；②基础理论及基本技能的掌握；③综合运用所学知识解决实际问题的能力；③工作量的大小；④取得的主要成果及创新点；⑤写作的规范程度；⑥总体评价及建议成绩；⑦存在问题；⑧是否同意答辩等）：

成绩： 评阅教师签字：

年 月 日

**中国矿业大学毕业设计（论文）答辩及综合成绩**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **答辩情况** | | | | | |
| 提出问题 | 回答问题 | | | | |
| 正确 | 基本正确 | 有一般性错误 | 有原则性错误 | 没有  回答 |
|  |  |  |  |  |  |
| 答辩委员会评语及建议成绩  答辩委员会主任签字：  年 月 日 | | | | | |
| 学院领导小组综合评定成绩  学院领导小组负责人：  年 月 日 | | | | | |

# 摘 要

随着移动通信技术的飞速发展，我们能够使用到越来越快速的网络，同时和功能更加强大的移动终端。作为饮食这样一个传统行业，我们一般都是投入较多的人力资源，这样无论是对顾客还是对餐饮企业来讲，效率都是比较低下的，而且成本也比较高。而让移动技术服务于餐饮行业，则是一个巨大的突破。

传统的餐饮行业中，餐厅点餐、送单结账都为人工，工作效率十分不高，并且在就餐高峰期人员繁杂的时候，服务员还有产生误操作的可能。通过前面的相关调研和分析，可以清楚的看到那种传统的工作模式已经越来越不适应这个飞速发展的行业了。为了解决这个问题，并且结合相关的技术,主要是手机和互联网技术，设计了一套基于传统的C/S架构的点餐系统，专为餐饮酒店服务。

该系统分为两个大部件，主要是客户端和服务器端。客户端设计为一个运行在当今特别流行的移动平台—android平台之上；而服务器端则采用相关的web技术来开发，主要是运行在Tomcat服务器上。最终可以实现的功能有如下几个方面。在客户端中，服务员可以为顾客开桌，顾客可以去了解菜品及价格，同时提供更新和结账等功能。而服务器端则是采用远程web登录，对后台进行管理。

本文将通过开发技术的展示，具体需求分析的介绍和详细设计的规划以及最终实现的方法来系统的概述该系统的开发流程。本系统主要采用Java语言作为主要的语言工具。数据库使用了MySQL以及android系统自带的SQLite。同时也利用了一些集成开发环境如Eclipse，MyEclipse等来进行开发。并运用相关的软件工具如Tomcat和Navicate来使得开发过程更为简便。

关键词：ANDROID；JAVA；eclipse；MySql；点餐系统

# ABSTRACT

With the rapid development of mobile communication technology, more quickly internent are availiable to us , and also more powerful mobile terminal. In traditional restaurant industry , we are generally hire more people, so both for the customer or for catering enterprises, the efficiency is relatively low , and the cost is relatively high. And let mobile technology services in the food and beverage industry is a huge breakthrough.

 Traditional restaurant industry , it is people to do everything they need to do, the working efficiency is not high , and at the top busy time , the waiter may produce the possibility of misuse . Through research and analysis related to the problem, you can clearly see that kind of traditional work patterns have become increasingly unsuited to the rapid development of the industry. To solve this problem, we use some related technology, mainly mobile phone and Internet technology , designed ordering system based on traditional C / S architecture , serviced for restaurant industry.

The system is divided into two major parts. Client is designed to run on top of a particularly popular in today's mobile platforms -android platform ; while the Server is using web technologies to program , mainly running on Tomcat server. The function of this system has the following aspects. In the client, the waiter can open tables for customers , customers can get to know the dishes and prices , while providing updates and billing functions. And the server is using remote web login in that administrator can do backstage management.

This article will show through the development of technology ,through introducing requirements analysis and detailed design . The system uses the Java language as the main language tools . Use the MySQL database and SQLite. Also the use of some integrated development environments such as Eclipse, MyEclipse , etc. to develop. And the use of related software tools such as Tomcat and Navicate to make the development process easier.

Keywords : ANDROID; JAVA; eclipse; MySql; online reservations system

目 录

1 绪论 1

1.1课题背景 1

1.2选题意义及开发目的概述 1

1.3本文主要工作 2

1.4论文结构 2

2 系统构建的相关环境和开发技术 3

**2.1系统体系结构** 3

2.1.1整体系统架构（C/S) 3

2.1.2服务器端系统架构 4

**2.2系统开发技术及工具介绍** 4

2.2.1 ANDROID 5

2.2.2数据库MySQL及SQLite简介 7

2.2.3UML面向对象建模工具 7

2.2.4　eclipse简介 8

2.2.5　Tomcat简介 8

2.2.4 Navicat For MySQL简介 9

2.3系统开发和运行环境 9

3 系统需求分析 10

3.1系统需求分析 10

3.1.1功能需求简要说明 10

3.1.2系统功能需求分析 10

3.2系统详细需求 11

3.2.1客户端需求分析 11

3.2.2 服务器端的需求分析 12

4概要设计 13

4.1模块的划分与功能设计 13

4.2功能模块的设计 13

4.3数据库结构的设计 13

5 系统详细设计 18

5.1 数据库设计 18

5.1.1数据库需求分析 18

5.1.2数据库需求分析 18

5.1.3数据库的逻辑设计 19

5.1.4连接数据库的设计与实现 21

5.2客户端设计 21

5.3服务器端的设计 23

5.4系统整体的业务流程 26

6 系统编码实现与测试 27

6.1系统实现的关键技术 27

6.2登录功能的实现 27

6.3 点菜功能的实现 30

6.4并桌功能的实现 33

6.5支付功能的实现 34

6.6更新，注销以及转台功能的展示 37

6.7系统功能测试 39

7 结 论 41

参考文献 42

翻译部分 43

英文原文 43

中文译文 47

致 谢 50

# 1 绪论

**1.1课题背景**

在餐饮行业中，传统的点菜方式基本全都是由人工来进行操作的。主要是服务员对顾客提出的要求进行相关的记录。这样在工作的时候很容易造成各种各样的麻烦和错误。例如由于手写字迹的潦草可能会导致认识上的差异，造成上菜错误。或者在传菜单的时候也可能造成遗忘和偏差，这些都是可能造成的错误。同时在顾客对自己的独特口味进行交代的时候，服务人员并不能准确的记录，容易造成偏差和遗忘。同时当客人催菜的时候也并能完全记得。会造成顾客就餐先后次序的混乱，造成不必要的麻烦。这些都是使用传统管理模式存在的弊端。因为这些漏洞的存在，使用现代的技术来对整个餐饮行业进行管理，可以改变这些陋习和误差，提高整个行业的服务水平，同时也使得顾客更加享受服务。

在当今这样的一个世界里，无线技术高度发展，无论是使用各种电话通信还是网络通信，都能够快速便捷的传递消息。点餐系统使用的传统的局域网布局，服务人员通过手持终端，将顾客的需求发给后台，后台在接收到相关的信息后就可以进行操作了。具体的操作就是对信息进行分类处理，把点菜信息传送给厨房，把订单信息传送给收银台。

## **1.2选题意义及开发目的概述**

这篇文章详细介绍了整个无线点餐系统的开发过程，论述了其中的方法和意义。在该系统的开发过程中，严格的遵守了软件工程开发方法的严谨性和系统性。首先文章会介绍系统的需求分析和基本的功能设计。接着会介绍详细设计以及实习的方法和最终的测试。而且，在开发之初，就深入调查和研究了整个餐饮行业的现状和管理方式，同时也调研了已经存在的点餐系统，分析其中的共性和个性，由此来得出该系统所要实现的功能以及具体的设计思路。最后介绍了整个系统的开发环境和方法以及数据库的设计与实现。同时还介绍了这些工具的使用方法。

系统的开发过程中，在概要设计这个阶段主要对后台数据库管理操作设计和客户端操作设计，同时介绍了数据库的建立和维护。详细设计主要阐述了点菜管理、并台管理、查台管理，转台管理，注销等以及后台管理功能的实现，并给出了系统的部分设计代码。

最后，在系统完全实现之后对系统进行了详尽的测试，最终的结果表明设计最终达到预期的目标，同时也能够很快速的响应各种请求。

* Android客户端的设计
* MySQL数据库的设计
* 后台服务器端的设计
* 系统总体设计
* 系统详细设计
* 程序调试及功能测试

## **1.3本文主要工作**

针对具体问题的分析和考察，该系统大概的可以分为两个方面，首先是客户端，即手持终端的设计。然后就是服务器端的设计了，主要是后台的相关设计和管理。体统采用了基于超文本传输协议的HTTP协议来进行数据和信息的传输。整体的架构采用的是C/S 架构。这种架构便于修改和部署，同时灵活多变，是一个比较经典的架构。客户端运行在基于android平台的移动终端之上，主要的功能性处理和信息的整理都有服务器端来完成，而客户端仅仅是一个输入输出的角色。并同时在客户端上进行菜品的展示，桌台的信息以及账单的结算。在服务器端可以进行相关的后台管理操作，例如菜单的修改、增删等等。

## **1.4论文结构**

第一章：绪论，主要是为了较少本系统的开发背景、发展现状、目的和意义等。

第二章：介绍了系统构建的相关环境和开发技术。

第三章：简要讲述了本系统设计前所做的需求分析等，阐明要做一个什么样的系统。

第四章：对系统的总体设计，勾勒设计的框架。

第五章：按用户类型对系统进行分模块的详细设计，是对需求分析的一个细化，包括点菜模块，开桌模块，换桌模块等。并初步设计封装类。

第六章：展示部分模块的编码实现，并进行简单的单体测试。

第七章：总述在系统设计前前后后的感受。

# 2 系统构建的相关环境和开发技术

**2.1系统体系结构**

系统的整体架构讲的是系统是怎样进行功能的划分和部署的。这个对系统选择什么样的架构是非常重要的。因为系统的架构直接决定了整个软件的开发难度和系统的性能。一个好的系统架构能够是的整个系统的性能变得十分的优秀。所以这个架构是必须认真考虑的一件事。

### 2.1.1整体系统架构****（C/S)****

总的来讲这个系统是用的架构是非常经典的一种架构，对于程序开发人员来讲，这中架构是十分熟悉的一种。通过这种架构可以很轻松的将软件的功能进行拆分。把不同的任务合理的分配到客户端和服务器端，通过这种办法可以降低系统的通讯开销，让整个系统对网络的需求降低，同时也减小网路负载。

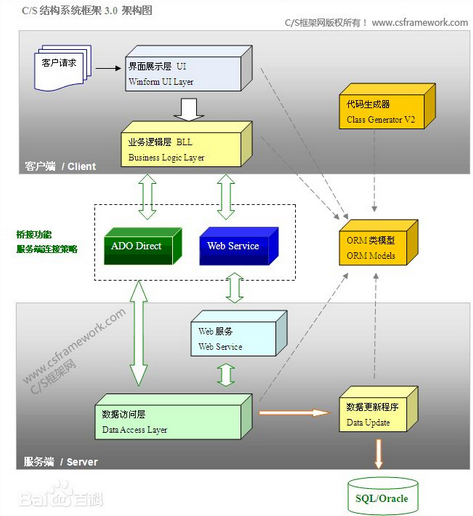


图2.1 C/S结构图

### 2.1.2服务器端系统架构

该系统的服务器端采用的是基于B/S结构（Browser/Server，浏览器/服务器模式）的架构。这种结构是一种网络结构，主要是用于在网络上的部署和开发。这种网络结构使用了统一的客户端，而让更多的功能在服务器的上面实现，这样便让客户端和服务器实行了分离，彼此之间更加独立。这样就会让服务器端和客户端的开发变得更为简单和快捷，这是因为彼此之间的独立性可以让彼此之间的代码变得高内聚，低耦合。只需要保证接口不变，就不要因为彼此的改变而去改变对方。同时又因为B/S结构最大的作用就是能够让程序在客户端的电脑上进行部分的处理，降低服务器的负担，从而进行局部的刷新和改善，使得系统的实时性和快捷性得到了很大的提升。另一方面，又使得数据的交换量变小，增强了系统整体的交互性。B/S结构图如图2-1所示：

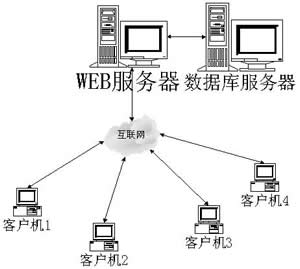


图2.2 B/S结构图

B/S结构软件有许多优势：

1. 使用B/S架构的软件成本是比较低的，并且有多种选择来部署自己的软件。 一般来讲，都会选择Tomcat服务器，因为该服务器在任何操作系统中都是免费的，而且该服务器的配置比较简单，所以客户会比较愿意选择这样的服务器。
2. 升级和维护方式便捷。如果B/S结构软件需要更新或者维护，维护人员只需要对服务器进行升级和维护。当服务器是部署在异地而管理和维护人员是在别的地方时，只需要通过互联网将服务器接入到其中，就可以直接进行相关的操作来实现远程维护和更新而不需要来回奔波，同时也可以节省用户人力和财力。

**2.2系统开发技术及工具介绍**

本节通过介绍ANDROID、数据库技术、JAVA,UML建模工具、eclipse，描述了系统开发所使用的具体架构和技术。阐明了所采用的各种技术的优点所在，并简要描述了UML建模语言。最后，详细的说明了系统具体开发环境和开发工具，明确了开发方式。

### 2.2.1 ANDROID

Android这个词语在英文中的本来含义是指的“机器人”的意思，但是现在Android却是作为一个移动平台的操作系统而广为人知。而其本身的含义估计很少有人知道吧。这主要是因为在2007年11月5日，Google宣布其基于Linux平台的开源手机操作系统的名称为Android。这个平台主要由以下几个方面构成，包括操作系统、中间件、用户界面和应用软件组成的。Google公司宣称这个是首个为移动终端真正打造的完全开放和完整的移动软件生态系统。

Android是一款软件平台，这款软件平台主要是基于Linux内核的。同时又是一款操作系统。早期主要有Google公司开发完成。Android在中国的注册商标是“安致”。因为其主要是一款开源项目，所以后来主要的开发工作就由开放手机联盟完成（Open Handset Alliance）。这个联盟是Google公司联合众多的硬件厂商组合而形成的。所以说能力是非常强大的。至于Android操作系统从出生到至今的版本很多，就不一一赘述了。Android的开发采用了软件堆层（software stack，又名以软件叠层）的架构，这种架构主要包含三个部分的内容。首先是底层的以Linux内核为基础的核心，仅仅是提供一些基础性的功能；其他的应用软件则是由各个公司自行开发并发行。主要是以Java为主要的编程开发语言。而由Google公司组建的联盟则有义务来推广该系统。目前除了众多厂家出的各种基于Android系统的手机外，Google公司本身也推出了基于该系统的平板和手机。而目前最新的SDK版本为本Android 4.4KitKat（奇巧巧克力）。

#### 1 Android Features特性：

Android系统有如下的几大特性[1]：

用于进行应用程序开发的基本框架，相关的组件，主要是用于重用和替换

一个名为Dalvik的虚拟机，这个虚拟机专门为移动设备进行了具体的优化

在Android系统内部中继承了一个浏览器，该浏览器是基于一个开源的WebKit引擎

优化的图形库 包括2D和3D图形库，3D图形库基于OpenGL ES

包含了针对该系统做了优化的2D和3D图形库，其中3D图形库是基于OpenGL ES

在系统中内嵌了一个数据库，名为SQLite，用作为结构化的数据存储

支持相关的多媒体文件，包括常见的音频和视频文件，以及静态映像文件之类的格式

GSM电话（依赖于硬件）

同时也支持蓝牙以及一些通信频段的制式如EDGE，3G，WIFI等等（依赖于硬件）

支持拍照和摄像功能，支持GPS，以及指南针和加速度测量仪等（依赖于硬件）

特别优良的开发环境，主要是其广泛的硬件支持，包括设备模拟器，调试工具，内存分析等等，都可以用来进行相关的测设和开发。同时特别的是Eclipse的集成开发环境

#### 2 Android基本框架（Android Architecture）

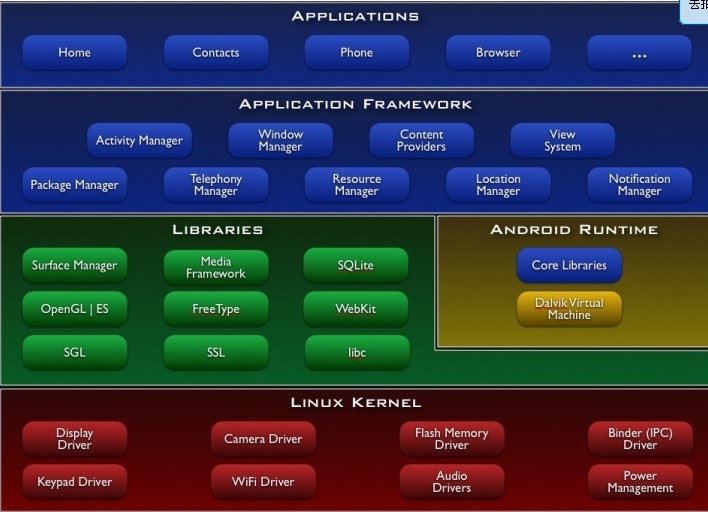
****

图 2.3 Android Architecture

#### 3 Android系统的四大组件

在Android操作系统之中，该系统为程序开发人员提供了四个基本的组件，这些个组件可以被看成是一个视图，就像在Windows下的窗口一样。但是在这些视图之中，有些是可以直接发现的，有些则是运行在后台并不可见的。例如Activity这个组件中，ContentProvider就可以直接被发现，而Service和BroadcastReceiver则是不可见的，因为他们运行在后台之中。下面将为大家分别介绍四大组件：

（1）Activity 简介

在Android手机操作系统之中，Activity为用户提供了一种人机交互的接口，即图形交互界面。通常，每一个Android应用程序基本上都是由很多个Activity组成的。每一个Activity都代表了一种功能和界面。这些Activity组成了自己的堆栈。也就是说在程序中功能的切换实际上就是堆栈的弹出与压入。这样就很容易理解当前活跃的Activity是在栈顶的。同时，由于该系统的特性，每一个Activity都有自己的生命周期，这个周期是由系统进行操控的。

（2）Service 简介

从Service这个单词本身就可以看出，这种就是运行在后台的一种为别的程序提供服务的服务程序。通常来讲，Service程序基本上不会和用户进行直接的交互，由于这种服务程序本身的特性，所以对这一类的程序来讲是没有可视化的界面。这样，定义一个Service类就变得比较简便了。程序开发人员只需要继承Service类，同时实现这种服务程序的生命周期方法。而当需要使用已经定义好的Service类的时候，就必须先在AndroidManifest.xml配置文件中对Service类进行注册，这种操作相当于一种声明，如此一来在使用时就可以直接用相关的函数来进行直接的操作了。

（3）Broadcast Receiver 简介

Broadcast Receiver这个服务机制脱胎于网络传输协议之上的广播监听协议。从这个词语的本身我们就可以知道是广播接收器。其功能和事件处理机制相类似，不同的地方是事件处理机制是程序组件级别的，就是用来处理具体的事物的，打个比方，就好比监听某个按键的单击事件。而广播事件处理机制则不仅仅局限于某一个具体的事物，该处理机制是系统级别的。直到目前为止，在Android系统中开发的程序，可以用Intent来启动一个程序组件，也可以使用一个叫做sendBroadcast()方法来启动一个系统级别的事件广播来进行消息的传输和传递。而在对应的接受程序之中，可以使用Broadcast Receiver来对这些消息进行监听，主要是监听这些广播的Intent。

（4）ContentProvider简介

Content Provider是一种存储数据的工具，这种存储数据的工具并不是单纯的为了数据的存储，它存在的主要目的是为了进行数据的共享。打个比方，拍照程序进行了拍照这样一个行为，生成了照片这样一种数据。而这个时候微博想要上穿拍摄的照片，这就需要进行数据的共享了。这种工具的存在主要是为了对不同的应用程序之间的数据互访提供了方便。在Android系统中，这种工具是跨应用共享数据的唯一办法。

在Android操作系统中，常用的数据类型比如音频和视频文件，图片文件，联系方式等等，都被提供了大量的Content Provider。一般来讲，这些都被定义在android.provider这样一个包的下面。通过这样的方法，只要是被这样定义过的数据都可以进行比较方便的访问，只要具备相关的权限就可以了。

### 2.2.2数据库MySQL及SQLite简介

无线点餐系统采用的是MySQL AB公司开发、发布并支持的一种开源数据库。即MySQL数据库。

MySQL数据库服务器支持多个用户，多个现成同时并发的运行。SQL（结构化查询语言）是一门数据库语言，因为其标准的特性和简单性，使得其成为世界上最流行的语言。而MySQL是一个相对于SQL　Serve以及Oracle而言，是一个比较精巧的系统结构。同时由于其开放源码的特性，强大的功能，灵活使用的方法，丰富的编程接口以及精妙的体系结构，使得很多的自由爱好者甚至是商业软件公司的喜爱。它成为了很多软件首选的数据库

SQLite是一种被设计为专为嵌入式移动平台使用的数据库，由此可以清晰的了解到该数据库是一种轻型的数据库。该数据库的开发严格遵守了ACID的相关规范，是一款关联式的数据库管理系统。就目前而言，这款数据库除了在Android系统中被使用着，还在Apple公司的IOS操作系统和其他的嵌入式产品中被广泛的使用着。由于其专门的设计方式，该数据库所占用的空间十分的低，对于目前以G为单位的内存来讲，它仅仅需要几百K的内存就足够了。该数据库能够支持Windows/Linux/Unix等等主流的[操作系统](http://baike.baidu.com/view/880.htm)。与此同时，这款数据库还能够和很多的编程语言相结合，并且拥有ODBC接口。SQLite从2000年诞生以来，到目前已经走过了十多个年头了，当前最新的版本为[3.8.4.3](http://www.sqlite.org/releaselog/3_8_4_3.html) 。

### 2.2.3UML面向对象建模工具

该点餐系统主要采用开发方法是基于面向对象的一种开发方法。主要是对整个系统中的相关需求进行过程抽象和具体功能模块的模块化。这种设计的方法强调的是利用各种图形来表达用户的需求。主要使用的一些方法有数据流、数据字典、结构化语言以及判定树等等在软件工程中进行了具体的规范的一些方法。主要使用的一种建模语言，该语言的具体情况如下。UML（Unified Modeling Language，统一建模工具），是一种基于面向对象方法的建模图形语言。这种建模语言主要是用来描述软件系统的的各个方面的，是一种面向对象的建模工具。UML在其中汇聚了软件工程的各种思想和技术，也便于描述开发过程的一种工具性的语言。总的来讲，这种语言功能相当的强大，可以很明确的定义各种需求设计，能够良好的表达各种东西，而且适用的方位还是比较的广泛的。该语言主要用于比较大型的系统工程的时候，可以是的整个开发的流程更加的直观和清晰。

在UML中大致的可以分为下面四种关系：关联、依赖、泛化以及实现。UML有两类图，分别是结构视图和行为视图。结构视图也称为静态模型图，主要强调系统的对象结构，而行为视图又称动态模型图，主要关注的是系统对象的行为动作。

主要运用以下UML中的五种类图在软件开发中进行建模：

* 1. 用例图。用例图主要是用来表示在系统中各个模块及角色之间的关系的。大多是角色和用例以及彼此之间的关系。用例是用来描述在系统之中，整个系统以及子系统和类的功能一致的集合。集中体现在系统和各个外部信息交换者的消息交换动作的次序。
  2. 协作图。这种图主要是用来讲述彼此协作的类和对象之间的交互关系和链接关系。体现了互相交互的对象之间的一种静态链接。
  3. 状态图。主要描述了在一些特定的条件下某些对象的所有可能状态及使得目前状态改变为特定状态现象，而这些条件是主要的。
  4. 活动图。主要用来讲述在整个系统的运行过程中需要进行的活动和这些活动依次发生的顺序。
  5. 顺序图。这种图主要是用来讲述系统中每个对象发送各种消息的顺序，同时还需要显示对象之间的交互联系，也就是说系统在执行到某一个特定时间需要发生的一些事件。

### 2.2.4　eclipse简介

Eclipse是一种功能十分强大的开发工具，基本上任何一种编程语言或者任何一种项目框架都可以使用Eclipse来进行开发。Eclipse之所以这么强大，是因为Eclipse可以随时的添加各种IDE来拓展自身的功能，这样就使得Eclipse基本上是无所不能的了。就本系统的开发而言，是使用的Eclipse IDE for Android Developers这样一种IDE，对于开发Android项目来讲是十分便利的。同时配置这样的环境也是十分的方便。

### 2.2.5　Tomcat简介

Tomcat这个是很受大部分的程序员的喜爱的一种服务器，造成这种现象的原因是这种服务器在运行的时候占用的系统资源是十分的少的。同时这种服务器能够被很好的扩展到更多的应用和平台之中。而且像负载平衡折后总功能以及邮件服务等等一些应用开发中常有的功能它都是支持的。与此同时，Tomcat还在不断的完善自己的各种功能。这让程序员们更加的喜欢它。

就其服务器本身的规模来讲，Tomcat是属于轻量级的服务器。这种服务器大多应用在中小型的项目和系统中。这主要是由于这类的系统和项目的访问人员并不是那么的多，所以会选择这样一个较小的服务器。

### 2.2.4 Navicat For MySQL简介

Navicat For MySQL是一种图形操作界面，主要是给MySQL数据库配上了一个交互界面可以直接进行图形化的操作，而不会直接在字符界面下进行操作。在这个软件下，直接连接到数据库之后就可以看到已经创建的数据库和建立的表了。同时也可以直接操作，不需要输入代码了。基本上一个数据库应该有的功能这个软件都是支持的。Navicate MySQL可以给程序员提供一种更为方便和快捷的形式来创建数据库，创建相关的表，组织信息和访问共享信息。

**2.3系统开发和运行环境**

* 该点餐系统使用了eclipse作为开发工具，采用Java作为系统开发语言。为了保证系统数据有强大的安全和稳定性。本系统数据库运用的是MySQL。
* 系统采用了B/S结构，结合三层架构，通过android ADT平台在Windows7系统中进行系统开发。
* 本系统采用android development环境为系统开发环境，android 模拟器和android手机作为调试工具。系统结合各类最新技术，加强了系统可用性和美观性

# 3 系统需求分析

**3.1系统需求分析**

需求分析是整个系统设计和实现之间的重要阶段。首先，需求分析必须按照实际的需求定义功能，并根据需要做定期的检查和修改。一个好的需要分析可以帮助我们早期就发现错误，让我们少走弯路，开发成本也会降低，开发软件起来更有效率。

### 3.1.1功能需求简要说明

该系统设计的主要目的是为了向发展到了一定规模并且经济条件较好的餐饮企业提供一个优良的电子菜单系统。通过将顾客对菜品的了解和点菜，酒店厨房加工菜肴，以及服务员的传菜等一系列的功能集成到这个系统之中。把每一个的环节都进行了明确的分工，并且提供了良好的人机交互界面，使得整个工作流程变得更为简便快捷，减少出错的概率，提供了精准的数字化服务，让餐饮行业的管理更加的规范化和合理化。同时也是传统行业和新兴技术行业相结合的一个典型。

### 3.1.2系统功能需求分析

  随着3G时代的到来，移动平台应用越来越普遍，为了使电子菜单的使用方便，特设计如下的功能模块

需求界面如下：

客户端：

1、登陆界面

2、订单下达界面

3、搜索菜品界面

4、订单查看界面

5、个人信息界面

6、菜品详情界面

7、结账界面

8、更新管理界面

服务器端：

1. 内容管理和添加

需求实现功能如下：

客户端：

1、实现客户端与服务器之间的连接

2、实现登陆功能

3、实现登陆后能浏览到服务器数据，并对数据进行一系列的操作

4、能把菜品加入到订单

5、把生成的订单信息传回服务器端

服务器端：

1、实现管理员登陆

2、实现添加、删除、修改菜品信息

3、实现对客户端传过来的订单进行处理

4、实现添加、删除会员列表

功能模块图如3.1所示：

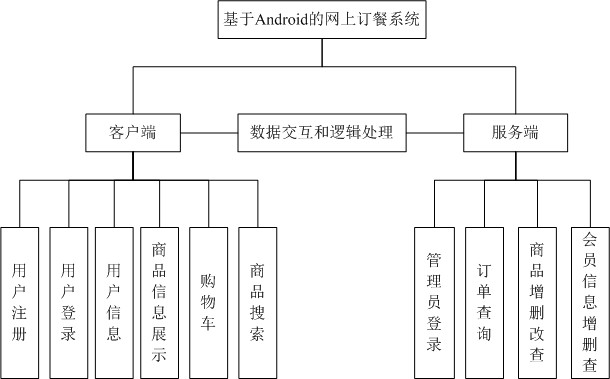


图3.1 系统功能模块图

## 3.2系统详细需求

### 3.2.1客户端需求分析

该客户端是设计在基于Android系统上的一个客户端。主要的功能包括登陆，点菜，开桌，并桌，查台，结账等等的功能。

#### 3.2.1.1 登录功能的需求分析

本系统在客户端登录的人员只有服务员这个唯一的角色。服务员自己不能在客户端注册，只能通过酒店的管理人员下发的账号和密码登陆。这个账号对应了唯一的服务员。这样便于管理整个酒店的服务人员的服务情况。

#### 3.2.1.2 点菜功能需求分析

在客户端中，当服务员进行登录之后，可以将手持终端交给顾客，顾客可以在点菜界面下进行相关的操作，也可以由服务员根据顾客的各种要求进行相关的操作。在点菜界面，首先要选择顾客所在的桌号，并将该桌号制空，说明这一桌有人了，当查台的时候就不可以再将这一桌开给别的客人。在选择完桌号之后，要进行点菜。这样可以直接进行查点。最后生成订单发往后台。

#### 3.2.1.3并台、查台和转台的需求分析

在顾客进行用餐的时候，可能会遇到熟人或者当客流量较大的时候，需要将较少人数的餐台并到一桌上，这时候就需要对桌台进行合并。这种合并需要服务员进行相关的操作。而当有人需要用餐时，服务人员需要对整个桌子的使用情况进行详细的了解。这就需要查台功能了。而当有特殊的情况发生时，有可能发生顾客需要换桌的情况，这就需要转台功能了。

#### 3.2.1.4更新和退出的功能

由于技术开发的有限，该系统并不能做到数据的实时更新，需有时不时的进行数据的更新。该功能的实现是通过手动的更新进行的，即需要服务人员经常将数据提交到服务器端。在进行完相关的操作之后，服务人员下班或者休息时，需要退出该系统。

### 3.2.2 服务器端的需求分析

服务器端的需求分析主要是对后台数据的业务逻辑处理。在后台服务器上，可以使用的有登录功能。这个登录功能是数据管理人员或者酒店管理人员才能够登陆的。登录的账号和密码写死在程序中，是唯一的。而后台管理人员登录之后就可以对服务人员进行管理，主要是增删和修改。同时也可以管理菜单的具体内容，包括添加菜肴，增减和修改价格等等一系列的功能。这些都可以直接通过后台管理体系进行具体的操作。

# 4概要设计

对一个正在进行开发的系统进行概要设计主要是为了对系统的整体功能进行详细的规划和设计。首先要将整个系统的功能模块进行具体的划分。主要是对各个功能模块具体的作用界限要有明确的规定和定义。同时还要考虑数据库的设计方式以及整个数据库的组织形式。这为系统接下来的详细设计做了较为重要的铺垫。在系统的设计过程之中起着一个很重要的作用。

## 4.1模块的划分与功能设计

这个系统的设计主要分成两个部分，包括客户端的设计和后台服务器的设计。在客户端上主要实现的是一些具体功能性的内容，主要的业务逻辑还是在后台进行具体的处理的。所以最重要的设计还是后台服务器端。

## 4.2功能模块的设计

在客户端系统中主要分成了如下的几个模块：包括点菜模块，这个模块是用来为顾客点餐的；结台模块，主要是用来结账和买单的；更新和注销等功能性的模块，这些模块主要是为了对应相关的操作。

结台——这个模块的具体设计是为了当顾客需要买单的时候，能够给顾客出具一份较为清晰的账单，方便顾客也方便酒店自己。

更新——这个模块是为了使当前的桌号和客户信息及时的和整个系统进行同步，更新服务器上的数据。

注销——在顾客结账以后进行的步骤，方便后来的客人能够及时了解酒店的信息。

## 4.3数据库结构的设计

在所有的系统工作过程中，数据库设计是十分重要的一个设计环节。而从规范的角度来讲，数据库设计 (Database Design)是指根据用户的需求，在根据这个系统的具体要求所建立的数据库中建立相关的数据库结构和数据表的过程。具体的来讲就是规划并且设计数据库中的数据对象的结构以及这些数据对象彼此之间的各种关系。

大致的来讲，数据库的设计可以分为一下几个部分：

1. 需求分析

在对数据库进行设计之前需要调查和分析用户的业务逻辑活动和数据的具体使用情况。同时还需要搞清楚各种数据的范围以及类型还有约束条件。要确定使用数据的各种条件，使得整个设计更为合理和严谨。

1. 概念设计

概念设计讲的是通过对整个系统的深入理解和对用户数据的深刻分析，从而得出的一些概念性的理论。通过对现实具体形象的了解，理解和分析，形成一个概念模型。主要是对现实世界的抽象。打个比方，在理解一个工厂或者一座学校的时候，不能单纯的至看到事情的表象，要深刻的理解。在工厂中，有工人，组长，车间主任，厂长，经理等人员，这些人员之间的关系需要抽象出来，是从属关系，多对多还直属关系等等。除此之外，还要看到厂房和员工之间的关系，都需要进行详尽的分析和抽象。这些都需要设计成E-R图，用来清晰的表示各种属性。

1. 逻辑设计

在这个设计过程中，主要的任务是将现实世界中的那些概念性的模型设计成在数据库中可以表示出来的一种逻辑模式，也就是说需要在数据库中能够适应于某种数据库管理系统所支持的逻辑数据模式。而且还要为这些数据在自身所对应的应用领域内产生出相对应的逻辑子模式。这个步骤实行完成之后，得到的结果就是所谓的“逻辑数据库”了。

1. 物理设计

在这个步骤中，一般来讲作为程序开发人员，很少会涉及到这样一步。因为这是数据库管理系统的开发人员需要完成的事情。在此简单的介绍一下，所谓的物理设计就是最终数据的存储都要归结在物理性质的存储介质上，在这个介质上用什么样的方式对数据进行存储，产生的效果和影响是完全不同，这一步的工作是对数据的物理存储方式进行设计，最终得到“物理数据库”。

1. 验证设计

在上述的设计完成之后，需要进行的工作就是对所设计的数据库进行相关的验证了。主要的验证方式就设就是对所涉及到的数据进行查验，并且进行测试，看看这些数据到底完整不完整。同时还需要验证这些数据之间的逻辑关系的完备性是否正确。并不断的测试，并且不断的修改。最终形成一个完整的数据库。为整个系统提供服务

在这个系统之中，创建的表主要有如下几种，将会通过数据库的截图来进行说明：

最终的数据库：

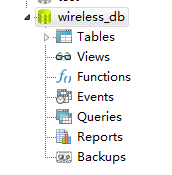


图4.2 数据库截图

系统数据库中的全部表如下表所示：

表4.3

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 表名称 | 表名含义 | 备注 |
| 1 | Checktable | 检查桌号 | 用来记录桌子的使用情况 |
| 2 | Menutb1 | 菜谱 | 记录菜单信息 |
| 3 | Menutypeid | 菜肴类型 | 记录是素菜还是荤菜 |
| 4 | Orderdetailtb1 | 订单详情 | 记录订单细节信息 |
| 5 | Ordertb1 | 订单 | 记录订单 |
| 6 | Queryorder | 查询订单 | 将订单查询结果放入该表 |
| 7 | Queryorderdetail | 查询详情 | 查询细节表 |
| 8 | Tabletb1 | 桌号表 | 记录桌号信息 |
| 9 | Usertb1 | 用户表 | 记录用户的信息 |

检查桌子是否为空的表定义如下：

表4.4 Checktable

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 宽度 | 备注 |
| 1 | Num | 桌号 | Int | 4 |  |
| 2 | Flag | 状态 | Int | 4 |  |

菜单表，用来存储菜单，定义如下：

表4.5 Menutb1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 宽度 | 备注 |
| 1 | Id | 编号 | Int | 4 | Primarykey |
| 2 | Price | 价格 | Int | 4 |  |
| 3 | Typeid | 菜肴类型 | Int | 4 |  |
| 4 | Pic | 图片位置 | varchar | 16 |  |
| 5 | Remark | 标记 | varchar | 16 |  |

菜肴类型表，用来记录菜肴的类型：

表4.6 Menutypeid

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 宽度 | 备注 |
| 1 | Id | 编号 | Int | 4 | Primarykey |
| 2 | Name | 菜肴类型 | varchar | 16 |  |

订单详情表，用来记录订单的详细情况：

表4.7 Orderdetailtb1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 宽度 | 备注 |
| 1 | Id | 编号 | Int | 4 | Primarykey |
| 2 | OrdereId | 订单号 | Int | 4 |  |
| 3 | Menuid | 菜品编号 | Int | 4 |  |
| 4 | Num | 菜品数量 | Int | 4 |  |
| 5 | Remark | 标记 | varchar | 16 |  |

订单表，用来记录订单的表：

表4.8 Ordertb1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 宽度 | 备注 |
| 1 | Id | 编号 | Int | 4 | Primarykey |
| 2 | OrdereTime | 订单时间 | Datetime | 8 |  |
| 3 | Userid | 用户id | Int | 4 |  |
| 4 | Tableid | 桌号 | Int | 4 |  |
| 5 | Personnum | 人数 | Int | 4 |  |
| 6 | Ispay | 付款状态 | Int | 4 |  |

查询表，用来记录查询的表：

表4.9 Queryorder

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 宽度 | 备注 |
| 1 | Ordertime | 订单时间 | Datatime | 8 |  |
| 2 | Name | 菜品类型 | Varcahr | 12 |  |
| 3 | Personnum | 人数 | Int | 4 |  |
| 4 | Tableid | 桌号 | Int | 4 |  |

查询细节表，定义如下：

表4.10 Queryorderdetail

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 宽度 | 备注 |
| 1 | Name | 菜品名称 | Varcahr | 12 |  |
| 2 | Price | 价格 | Int | 4 |  |
| 3 | Num | 数量 | Int | 4 |  |
| 4 | Total | 总价 | Int | 4 |  |
| 5 | Remark | 标记 | varchar | 16 |  |

桌号表，记录桌子的情况：

表4.11 Tabletb1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 宽度 | 备注 |
| 1 | Id | 桌号 | Int | 4 | Primary |
| 2 | Num | 数量 | Int | 4 |  |
| 3 | Flag | 标识 | Int | 4 |  |
| 4 | Description | 描述 | Varchar | 12 |  |

用户表，记录当前允许的用户：

表4.12 Usertb1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 宽度 | 备注 |
| 1 | Id | 编号 | Int | 4 | Primarykey |
| 2 | Account | 账户 | Int | 4 |  |
| 3 | Password | 密码 | Int | 4 |  |
| 4 | Name | 姓名 | Varchar | 12 |  |
| 5 | Gender | 性别 | Varchar | 12 |  |
| 6 | Permission | 权限 | Int | 4 |  |
| 7 | Remark | 描述 | Varchar | 12 |  |

# 5 系统详细设计

## 5.1 数据库设计

### 5.1.1数据库需求分析

整个无线点餐系统主要的服务对象是餐饮行业，所以其中的信息大多是餐饮行业的数据和信息。因此在这个系统之中的数据库必须和这个行业所对应的数据有着密切的关系。首先在这个数据库之中必须有用来保存菜单信息的一个表，这个表的主键采用的是编号，这样方便于快速的检索和查询。同时还要有保存服务人员信息的表，这个表中所要包含的信息有如下几点。首先帐号和密码，这个是服务人员的身份认证的标识，是唯一对应的。同时还要有服务人员的基本信息，如姓名，性别，年龄，权限等等。都需要进行详细的考虑。此外还要有保存酒店桌号信息的表，这样的表是用来记录桌子的信息，以便酒店服务人员和管理人员进行相关的查询，提供便利。除此之外，还要有记录订单的表，用以保存顾客点餐的详细记录，方便后台厨房及时进行加工，也方便结账时候的结算。还有一些别的用来进行辅助存储的表，就不一一赘述了。

### 5.1.2数据库需求分析

根据整个系统的分析，画出了如下的E-R图，这个图是用来表示整个数据库的关系的图：

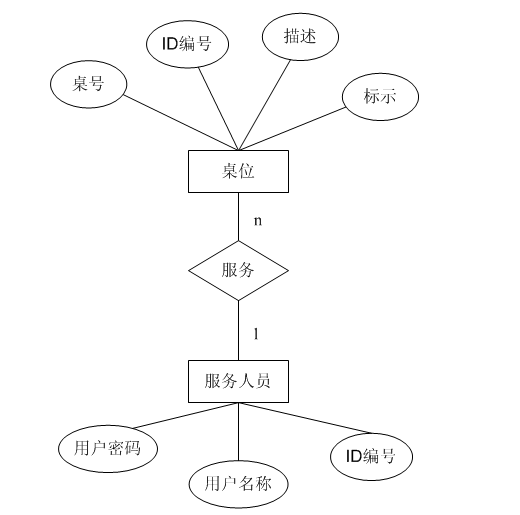


图5.1实体联系图

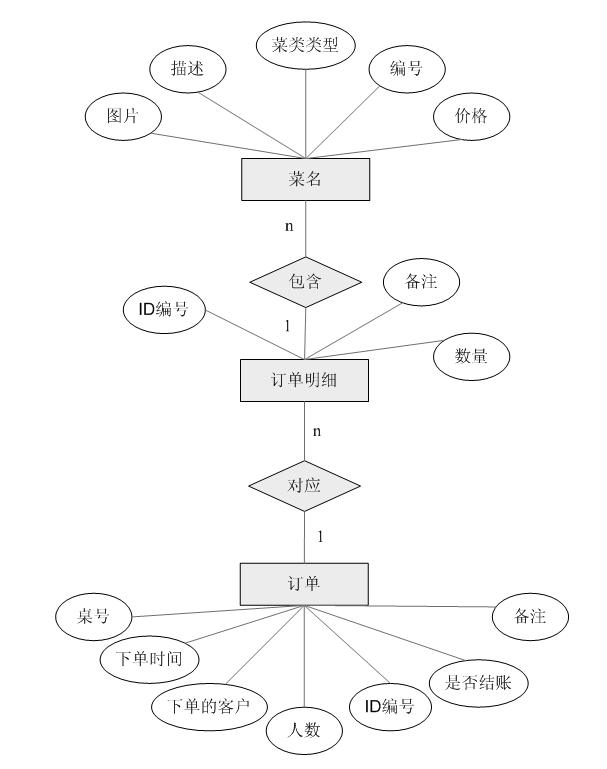


图5.2实体联系图

### 5.1.3数据库的逻辑设计

在进行数据库的逻辑设计这一个步骤之中，主要的任务是将从现实世界中抽象出来的概念数据库模型转换为具体的，依赖于具体数据库管理系统的逻辑数据库模型。而根据无线点餐系统的需求分析，以及进行的相关的数据库的概念设计，主要将系统的存储设计为如下的几个表，这些表格是代表的整个系统存储的主要关系

具体的说明如下：

（1）桌号表

桌号表的结构及相关说明如表5.1所示。

表5.1 桌号表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 宽度 | 备注 |
| 1 | Id | 编号 | Int | 4 | Primarykey |
| 2 | number | 桌位号 | Int | 4 |  |
| 3 | flag | 状态位 | Int | 4 |  |
| 4 | description | 描述 | varchar | 16 |  |

（2）订单表

订单表的结构及相关说明如表5.2所示。

表5.2 订单表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 宽度 | 备注 |
| 1 | Id | 订单号 | Int | 4 | Primarykey |
| 2 | orderTime | 下订单时间 | Int | 4 |  |
| 3 | userID | 服务员号 | varchar | 16 |  |
| 4 | tableId | 桌位号 | Int | 4 | Foreign key |
| 5 | personNum | 顾客人数 | Int | 4 |  |
| 6 | isPay | 是否付款 | varchar | 16 |  |
| 7 | remark | 备注 | varchar | 16 |  |

（3）订单详细表

订单详细表的结构及相关说明如表5.3 所示。

表5.3订单详细表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 宽度 | 备注 |
| 1 | Id | 编号 | Int | 4 | Primarykey |
| 2 | orderId | 下订单号 | varchar | 16 | Foreign key |
| 3 | menuId | 菜谱号 | Int | 4 | Foreign key |
| 4 | number | 点菜数 | Int | 4 |  |
| 5 | remark | 备注 | varchar | 16 |  |

### 5.1.4连接数据库的设计与实现

本系统采用java语言编写，使用jdbc连接Mysql数据库，所以要用到java中的connection对象和mysql驱动连接到数据库。

在实现该系统的时候，所采用的语言是Java语言，因此需要采用适合Java语言的数据库连接驱动来实现，也就是用JDBC驱动来连接MySQL数据库。同时还要用到Java中的connection对象以及MySQL中的驱动，这样和数据库交互就大致形成了。具体的代码如下所示：

String driver=com.mysql.jdbc.Driver

String url=jdbc:mysql://localhost:3360/wireless\_db?

useUnicode=true&characterEvcoding=utf-8

String username=root

String password=

Connection conn;

Class.forName(driver);

Conn=DriverManager.getConnection(url,username,password)

只有成功连接到了数据库才能进行其他操作，否则就无法开展以下的设计与实现工作了。因此，作为程序的开发人员，应该十分清晰的认识到连接好数据库是多么的重要。

## 5.2客户端设计

客户端的设计主要是为了是顾客和服务人员能够快速的学习和使用该系统，并需要花费很大的精力，所以客户端的设计必须十分的人机友好。作为一个客户端，是用户直接接触到的软件产品，所以设计上和实现上必须优良精巧。在整个客户端进行设计的时候，首先是登录界面。整个界面设计的十分的简洁，只有用户名，用户密码，登录和取消等按钮。这个设计让别人能够十分清楚的认识到这个界面就是用来进行身份的认证的。

在通过身份的认证之后，整个客户端的功能界面就展现在用户的面前了。通过整个界面，就能很清楚的了解这个客户端的具体的功能，这些功能一一对应这这些按钮的图标。直接读取图标上的文字，就可以知道这个图标的具体功能了。接下来将通过对这些具体功能设计的方案进行展示。

首先介绍的是登录界面的设计方案：

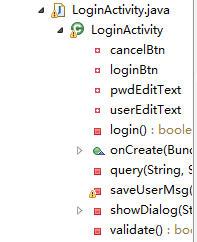


图5.3 Android端登录Activity布局结构

该模块的流程如如下图：



图5.4 流程图

然后介绍的是登录后的主菜单的布局结构：

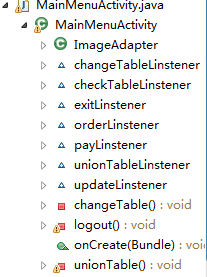


图5.5 Android端主界面Activity布局结构

然后介绍一下点菜的流程图：



图5.6 点菜流程图

从这些布局中可以清晰的了解诶整个客户端程序的设计，接下里的各个功能模块就不赘述了。

## **5.3**服务器端的设计

整个服务器端的设计需要进行的工作就是对所有的数据进行后台的处理。主要的完成工作就是进行业务逻辑的设计。在整个服务器端除了进行相关的业务逻辑处理，还需要完成相关页面的设计与处理，主要的工作就是设计登陆模块，修改模块等等。在服务器端的具体设计可以参考下面的业务逻辑图：

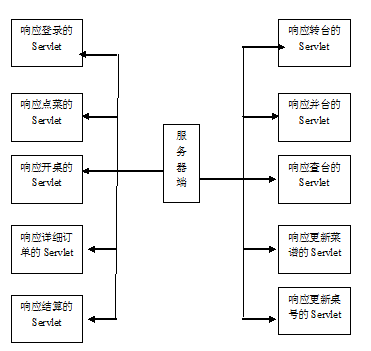


图5.7服务器端业务逻辑结构

在整个服务器端主要是通过Servlet来进行通信的，具体的设计如下图：

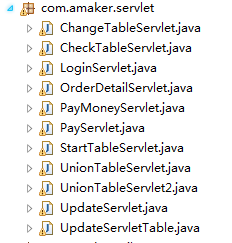


图5.8 服务器端Servlet设计结构

同样的需要展示的是在服务器端是如何响应客户端的请求的。同样将通过图片来介绍各个页面的主要逻辑设计结构。

首先介绍的是登录响应模块的设计：

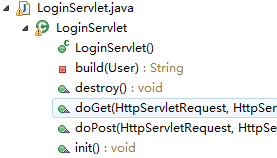


图5.9服务器端登录的Servlet设计结构

下图是服务器响应登录模块的流程图



图5.10 服务器响应登录流程图

然后是点菜功能模块的布局：

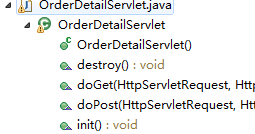


图5.11 服务器端点菜的Servlet设计结构

别的功能模块的设计大体是相同的。总的来讲在整个服务器的设计就是响应客户的各种请求以及对客户端发来的数据进行及时的处理。

## **5.4系统整**体的业务流程

对于整个系统来讲，基本上最终实现的功能是数据的提交和处理。所以整个流程图具体如下



图5.12 整体系统流程图

由上图可见，整个系统就是客户端充当数据的输入和输出端，而服务器充当处理的中心。总的来讲，就是首先由客户端提交用户名和密码，服务器端将接受到的信息和数据库进行匹配。若匹配成功，则将登录成功的信息进行返回，否则返回登录失败。当登录成功之后，系统则会进入主界面。在主界面上，会显示出该系统所具有的所有的功能。通过对对应功能的操作，则可以实现相关的需求。

# 6 系统编码实现与测试

## 6.1系统实现的关键技术

在整个系统的实现过程中，几乎没有使用什么技术。就客户端而言，客户端就是一个纯粹的Android软件，主要负责的工作就是和服务器联系，进行数据输入与输出。关键的技术就是在服务器端的设计中使用到的，那就是Servlet技术了。对于Servlet技术，与其讲这个是一种技术倒不如说这其实是一个程序，只不过这个程序是运行在服务器上的。这个小程序的主要作用是在一些系统中，需要实现访问数据库，进行数据的传输和处理，这时候就需要使用Servlet技术来对这些数据进行传送。使用这样一种的技术，可以使得整个系统的开发变得十分的简便和快捷，更加方便的实现整个系统。

对于Servlet技术的交互过程大致如下：首先客户端将一个请求发送给服务器端，而服务器在接受到请求之后先将这个请求发送给Servlet，通过Servlet去处理这个请求，生成对应的响应信息，然后Servlet再将响应的内容传送给服务器，服务器在接收到这个响应之后在将其发给客户端。具体的流程就是这个样子，总的来讲是很容易理解的。

## 6.2登录功能的实现

整个登录功能的实现就是为了进行身份验证，这样就可以在对登录人员进行限制，避免出现身份不明确的人员使用这个系统。主要是对登录帐号和密码进行验证。

登录界面的截图如下：



图6.1用户登录图

为这个登录功能的实现，程序开发人员单独建立了一个Activity来控制用户的登录。然后当用户输入完帐号和密码后，系统将先后判断帐号和密码的输入是否为空的字符，如果是则会提升用户输入相关的信息。当判断不为空的时候，系统则会连接到数据库访问数据库中的用户信息表，将帐号和密码进行匹配，如果匹配成功，则说明身份的认证通过了，则将进入下一个界面，否则认证失败，退出系统。

实现登录主要代码及分析：

// 登录方法

private boolean login(){

// 获得用户名称

String username = userEditText.getText().toString();

// 获得密码

String pwd = pwdEditText.getText().toString();

// 获得登录结果

String result=query(username,pwd);

if(result!=null&&result.equals("0")){

return false;

}else{

saveUserMsg(result);

return true;

}

在服务器端中相关的数据传送是通过Servlet来实现的，具体的实现方法如下：

public class LoginServlet extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.setContentType("text/html");

PrintWriter out = response.getWriter();

UserDao dao = new UserDaoImpl();

// 获得客户端请求参数

String username = request.getParameter("account");

String password = request.getParameter("password");

User u = dao.login(username, password);

if(u!=null){

// 响应客户端内容，登录成功

out.print(build(u));

}else{

// 响应客户端内容，登录失败

out.print(build(u));

// out.print("0");

}

out.flush();

out.close();

}

当通过身份验证后就进入主界面了，主界面的实现截图如下：

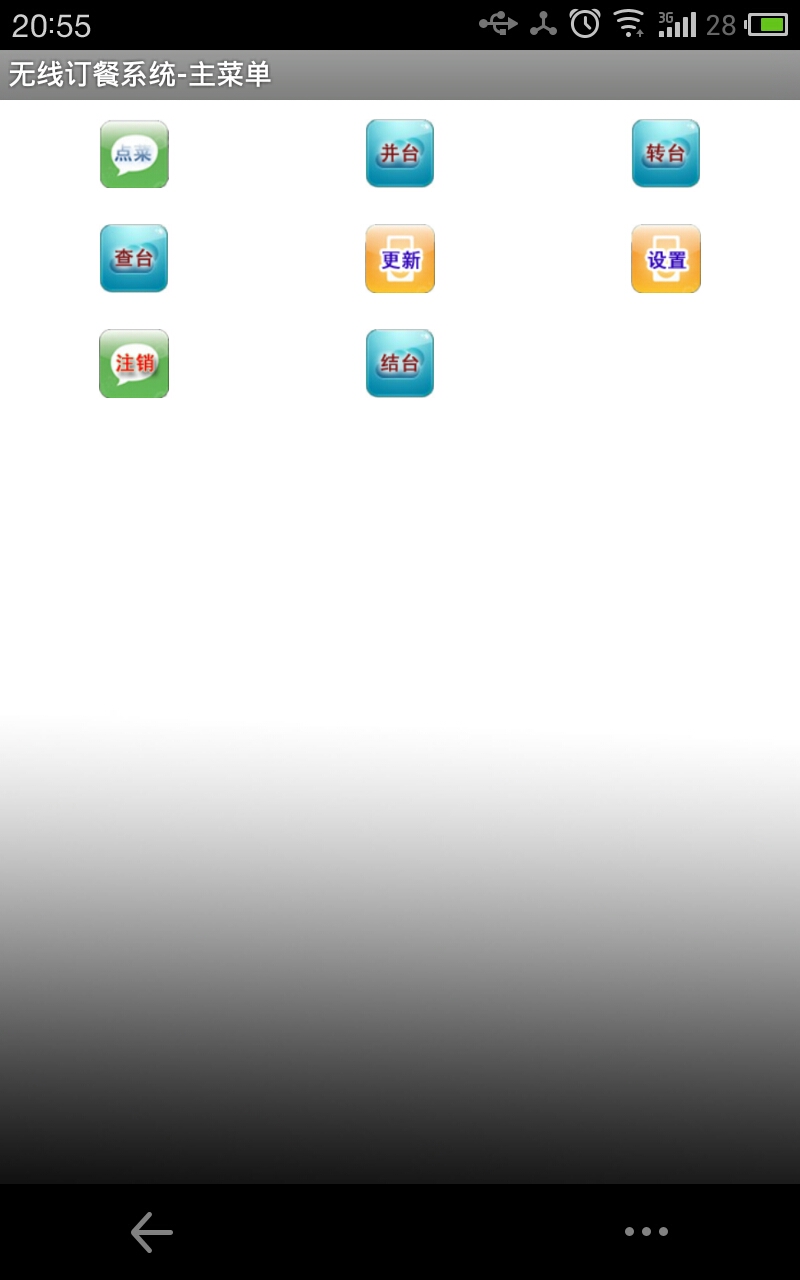


图6.2主界面图

通过这个界面就可以对系统进行具体的操作了。

## 6.3 点菜功能的实现

点菜的功能主要为展示菜品，开桌，以及下单。实现的界面如下图

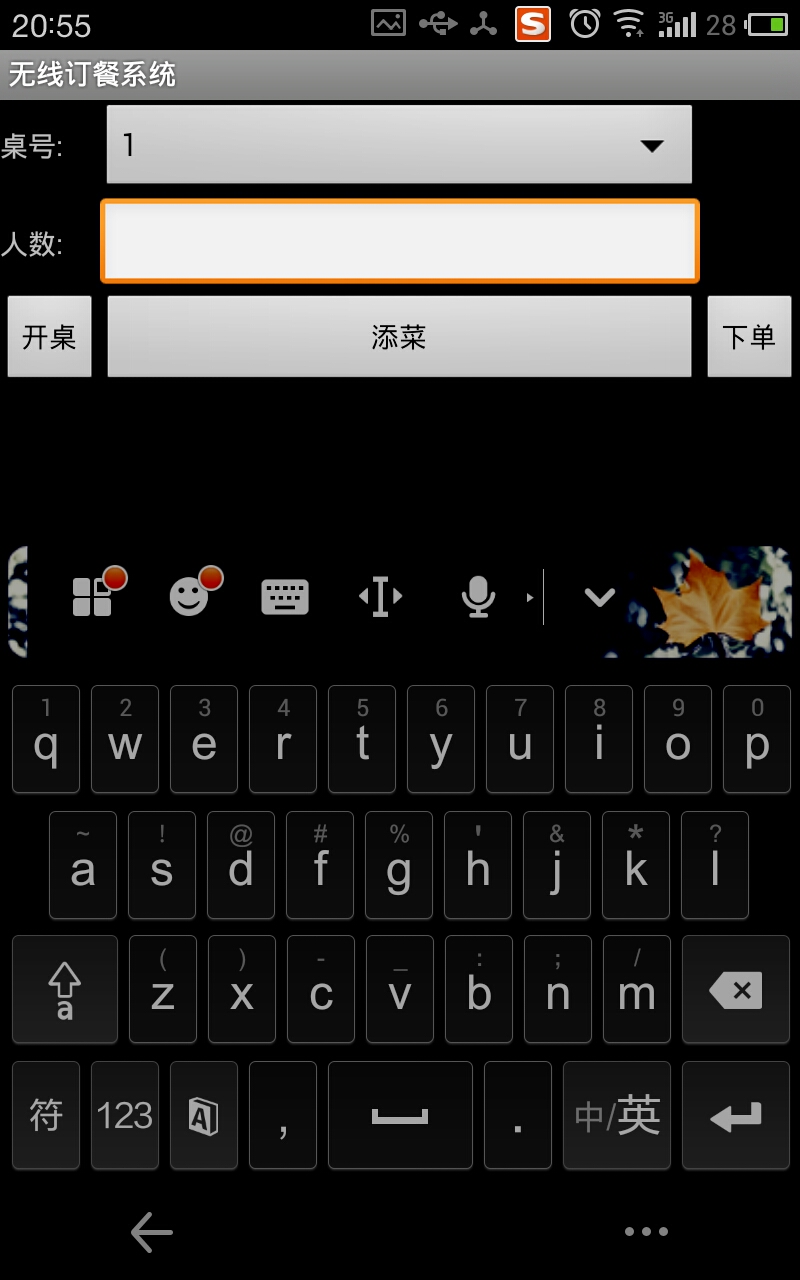


图6.3订单的界面图

在这个界面上可以下单，点菜和开桌。

在这个界面上的实现实际是从数据库中获取相关的信息。这里信息的获取并不是通过URL直接访问数据库的。而是首先在Android系统自带的数据库SQLite中创建了对应的数据库和两张表。一张表是用来存储桌子情况的表，另一张表是用来将数据库中的菜单信息存储到本地的数据库中，这样减轻了服务器的压力，也降低了客户端对服务器的依赖程度。

首先是设置Activity的布局以及设置相应的按钮监听器，具体代码如下：

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

// 为Activity设置界面布局

setContentView(R.layout.order);

// 实例化Spinner

tableNoSpinner = (Spinner) findViewById(R.id.tableNoSpinner);

// 为桌号下拉列表Spinner绑定数据

setTableAdapter();

// 实例化开桌按钮

startBtn = (Button) findViewById(R.id.startButton02);

// 为开桌按钮添加监听器

startBtn.setOnClickListener(startListener);

// 实例化点菜按钮

addBtn = (Button) findViewById(R.id.addMealButton01);

// 为点菜按钮添加监听器

addBtn.setOnClickListener(addListener);

// 实例化下单按钮

orderBtn = (Button) findViewById(R.id.orderButton02);

// 为下单按钮添加监听器

orderBtn.setOnClickListener(orderListener);

// 实例化人数编辑框

personNumEt = (EditText) findViewById(R.id.personNumEditText02);

// 实例化ListView

lv = (ListView) findViewById(R.id.orderDetailListView01);

// 为点菜列表ListView绑定数据

setMenusAdapter();

}

而在下来列表中则使用的是Spinner方法，具体的方法如下：

// 为桌号下拉列表Spinner绑定数据

private void setTableAdapter(){

// 访问本地SQLite数据库中桌号表的

System.out.println("更新下拉列表");

Uri uri = Uri.parse("content://com.amaker.provider.TABLES/table");

// 要选择桌号表中的列

String[] projection = { "\_id", "num", "description" };

// 查询放回游标

@SuppressWarnings("deprecation")

Cursor c =null;

c=managedQuery(uri, projection, null, null, null);

if(c!=null){

System.out.println("获取数据成功");

}else{

System.out.println("获取数据失败");

}

// 实例化桌号下拉列表Spinner的Adapter

@SuppressWarnings("deprecation")

SimpleCursorAdapter adapter2 = new SimpleCursorAdapter(this,

android.R.layout.simple\_spinner\_item, c,

new String[] { "\_id" }, new int[] { android.R.id.text1 });

// 为桌号Spinner绑定数据

tableNoSpinner.setAdapter(adapter2);

}

## 6.4并桌功能的实现

并桌功能模块主要是为了将几桌的顾客合并到一桌上。设计形成的界面如下：



图6.4并桌界面图

在这个界面中可以合并两桌的客人。

## 6.5支付功能的实现

支付功能是需要读取订单的信息，同时生成账单，让顾客进行支付。同时还要将顾客使用的桌子重新置为空。页面如下图：

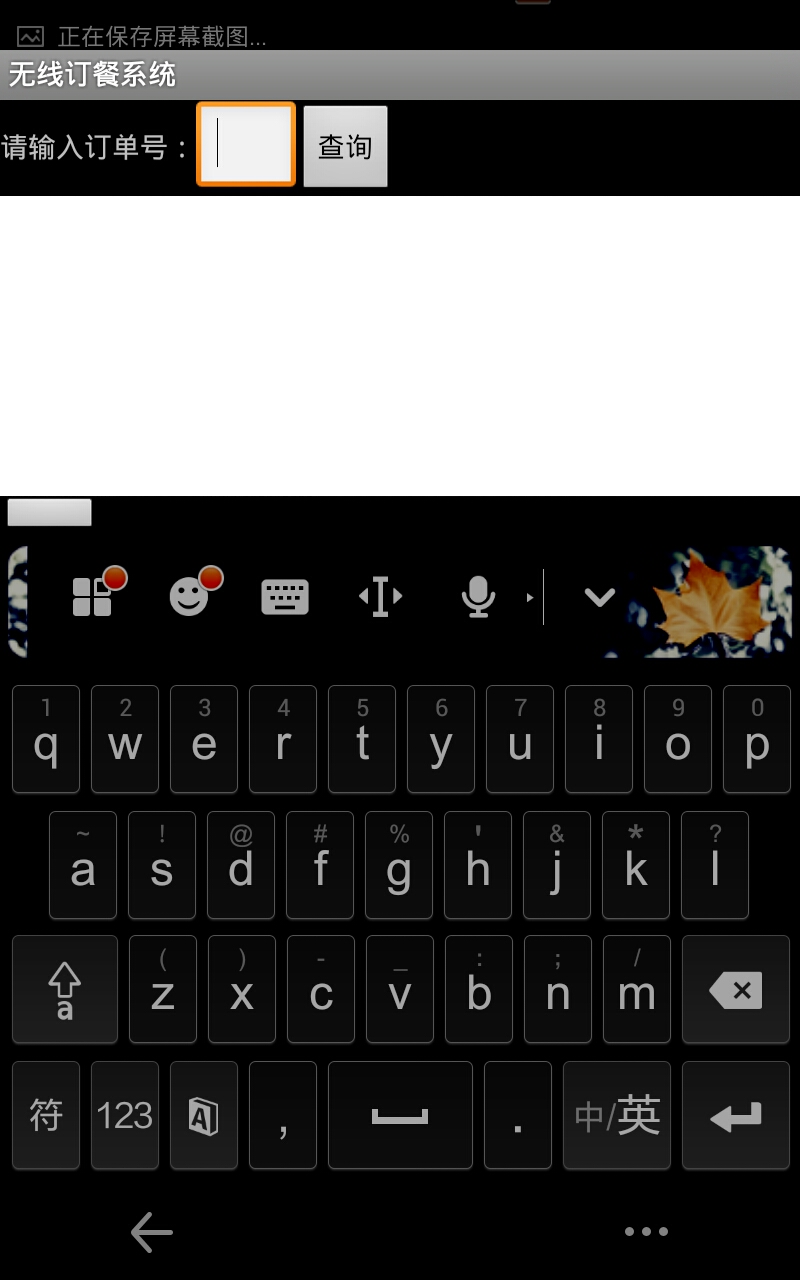


图6.5结台界面图

具体的代码如下：

// 查询点餐信息监听器

OnClickListener queryListener = new OnClickListener() {

@Override

public void onClick(View v) {

// 获得订单编号

String orderId = orderIdEt.getText().toString();

// 请求服务器url

String url = HttpUtil.BASE\_URL+"servlet/PayServlet?id="+orderId;

// 将返回信息在WebView中显示

wv.loadUrl(url);

}

};

// 结算监听器

OnClickListener payListener = new OnClickListener() {

@Override

public void onClick(View v) {

// 获得订单编号

String orderId = orderIdEt.getText().toString();

// 请求服务器url

String url = HttpUtil.BASE\_URL+"servlet/PayMoneyServlet?id="+orderId;

// 获得查询结果

String result = HttpUtil.queryStringForPost(url);

// 显示结算结果

Toast.makeText(PayActivity.this, result, Toast.LENGTH\_LONG).show();

// 使结算按钮失效

payBtn.setEnabled(false);

}

};

}

在服务器端的实现代码如下：

public class PayServlet extends HttpServlet {

// 构造方法

public PayServlet() {

super();

}

// 销毁方法

public void destroy() {

super.destroy();

}

// 响应Get请求

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.setContentType("text/html;charset=utf-8");

PrintWriter out = response.getWriter();

// 实例化DAO

PayDao dao = new PayDaoImpl();

// 获得订单ID

String id = request.getParameter("id");

// 查询订单信息

QueryOrder qo = dao.getOrderById(Integer.parseInt(id));

// 查询订单详细列表

List list = dao.getOrderDetailList(Integer.parseInt(id));

// 拼HTML页面展示

out

.println("<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.01 Transitional//EN\">");

out.println("<HTML>");

out.println(" <HEAD></HEAD>");

out.println(" <BODY>");

out.print("<table>");

out.print("<tr>");

for (int i = 0; i < list.size(); i++) {

QueryOrderDetail qod = (QueryOrderDetail) list.get(i);

String name = qod.getName();

int price = qod.getPrice();

int num = qod.getNum();

int total = qod.getTotal();

String remark = qod.getRemark();

// 响应Post请求

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doGet(request,response);

}

## 6.6更新，注销以及转台功能的展示

更新界面的截图：



图6.6更新界面图

注销界面截图：



图6.7注销界面图

转台界面截图：



图6.8转台界面图

## 6.7系统功能测试

测试，主要描述一种用来促进鉴定软件的正确性、完整性、安全性和质量的过程。软件测试的经典定义是：在规定的条件下对程序进行操作，以发现程序错误，衡量软件质量，并对其是否能满足设计要求进行评估的过程。

1用户登录测试

|  |  |
| --- | --- |
| **测试步骤** | **预期结果** |
| 输入正确的用户名、密码。 | 登录成功。 |
| 输入不存在的用户名 | 提示用户名不存在。 |
| 输入正确的用户名错误的密码 | 提示密码错误。 |

2点菜测试

|  |  |
| --- | --- |
| **测试步骤** | **预期结果** |
| 点击桌号下拉菜单 | 出现预先设置的并且空闲桌子 |
| 点击添菜按钮 | 出现存储的菜品 |
| 点击下单按钮 | 下单成功 |

3更新测试

|  |  |
| --- | --- |
| **测试步骤** | **预期结果** |
| 点击更新菜品按钮 | 本地菜品表和服务器端一致 |
| 点击更新桌号按钮 | 最新的桌子状态更新到本地 |

4结账功能测试

|  |  |
| --- | --- |
| **测试步骤** | **预期结果** |
| 输入存在的订单号 | 显示具体的信息并给出价格 |
| 输入不存在的订单号 | 提示输入错误 |
| 输入已经结过账的订单号 | 提示该订单已经结账 |

# 7 结 论

通过对这样一个课题的研究，让我对Android系统的理解更加的深入。特别是对于Android开发的熟悉，让我在今后的工作中能够更好的操纵对于Android应用程序的开发。同时也了认识到了系统的软件开发是非常重要的。软件工程并不仅仅是一种理论。

同时也对Java开发语言和层次开发有了更深刻的认识。通过分层实现系统。其优点在于使项目结构更清楚，分工更明确，有利于后期的维护和升级。它未必会提升性能，因为当子程序模块未执行结束时，主程序模块只能处于等待状态。这说明将应用程序划分层次，会带来其执行速度上的一些损失。但从团队开发效率角度上来讲却可以感受到大不相同的效果。

最后，最让我感受深刻的是对于设计模式的使用，在刚刚接触设计模式的时候，个人认为设计模式完全没有用，在对这个系统进行开发的时候，让我深刻理解了设计模式的重要性。同时代码的重用也是提高开发效率的一个途径。

参考文献

[1] 张孝祥著． java就业培训教程[M]．北京:清华大学出版社，2011

[2] 余志龙编著．Google Android SDK开发范例大全[M]．北京:人民邮电出版社，2012

[3] E2ECloud工作室著．深入浅出Google Android[M]．北京:人民邮电出版社，2010

[4] 靳岩 姚尚朗著．Google Android开发入门与实战[M]．北京:清华大学出版社，2011

[5] 高昂著．支持动态语言的Android平台[M]．北京:人民邮电出版社，2009

[6] eoe.Android开发社区．http://www.eoeandroid.com/． 2010-04-10.

[7] JavaEye论坛．http://www.javaeye.com/forums． 2010-04-15.

[8] Android API文档．http://androidappdocs.appspot.com/index.html． 2010-04-10.

[9] 高焕堂编．Android 应用框架原理与程式设计（第三版）[M]．北京:清华大学出版社，2011

[10] 杨丰盛主编．Android 应用开发揭秘[M]．西安:机械工业出版社，2010

[11]  萨师煊 王珊．数据库系统概论[M].北京：高等教育出版社,2005年

[12]  余志龙 郑名杰等．Google.Android.SDK开发范例大全[M].北京：人民邮电出版社,2011年

[13]  张海藩．软件工程导论[M]．北京：清华出版社，2003年

[14]  墨菲 李雪飞 吴明晖译．Android开发入门教程[J].北京：人民邮电出版社,2010年

[15]  李宁．Android开发权威指南[J].北京：机械工业出版社.2011年.

[16]  王向辉，张国印，沈洁．Android应用程序开发[M].北京：清华大学出版社，2010

[17]  普雷斯曼．软件工程实践的研究方法(第一版)[OL]. 西安：机械工业出版社，2011

[18]  Michael Otey,Paul Conte，陈恩义 吴强译．SQL Server开发指南[M].北京：清华大学出版社，2011

[19]  Ed Burnette．Hello，Android[OL]. UK： Pragmatic Bookshelf出版社.2009年

[20]  Reto Meier．Professional Android 2 Application Development[OL]. US：Wrox出版社，2010年

[21]  林锐．软件工程思想[M]，2000年

翻译部分

## 英文原文

Understand Android Security

The next generation of open operating systems won’t be on desktops or mainframes but on the small mobile devices we carry every day. The openness of these new environments will lead to new applications and markets and will enable greater integration with existing online services.

However, as the importance of the data and services our cell phones support increases, so too do the opportunities for vulnerability. It’s essential that this next generation of platforms provide a comprehensive and usable security infrastructure.Developed by the Open Handset Alliance (visibly led by Google), Android is a widely anticipated open source operating system for mobile devices that provides a base operating system, an application middleware layer, a Java software development kit (SDK), and a collection of system applications. Although the Android SDK has been available since late 2007, the frst publicly available Android-ready “G1” phone debuted in late October 2008. Since then, Android’s

growth has been phenomenal: TMobile’s G1 manufacturer HTC estimates shipment volumes of more than 1 million phones by the end of 2008, and industry insiders expect public adoption to increase steeply in 2009. Many other cell phone providers have either promised or plan to support it in the near future.

A large community of developers has organized around Android, and many new products and applications are now available for it. One of Android’s chief selling points is that it lets developers seamlessly .

extend online services to phones. The most visible example of this feature is—unsurprisingly—the tight integration of Google’s Gmail, Calendar, and Contacts Web applications with system utilities. Android users simply supply a username and password, and their phones automatically synchronize with Google services. Other vendors are rapidly adapting their existing instant messaging, social networks, and gaming services to Android, and many enterprises are looking for ways to integrate their own internal operations (such as inventory management, purchasing, receiving, and so forth) into it as well.Traditional desktop and server operating systems have struggled to securely integrate such personal and business applications and services on a single platform; although doing so on a mobile platform such as Android remains nontrivial, many researchers hope it provides a clean slate devoid of the complications that legacy software can cause. Android doesn’t ofcially support applications eloped for other platforms: applications execute on top of a Java middleware layer running on an embedded Linux kernel, so developers wishing to port their application to Android must use its custom user interface environment. Additionally, Android restricts application interaction to its special APIs by running each application as its own user identity. Although this controlled interaction has several benefcial security features, our experiences developing Android applications have revealed that designing secure forward. Android uses a simple permission label assignment model to restrict access to resources and other applications, but for reasons of necessity and convenience, its designers have added several potentially confusing refnements as the system has evolved.This article attempts to unmask the complexity of Android security and note some possible development pitfalls that occur when defning an application’s security. We conclude by attempting to draw some lessons and identify opportunities for future enhancements that should aid in clarity and correctness.Android Applications The Android application framework forces a structure on developers. It doesn’t have a main() function or single entry point for execution—instead, developers must design applications in terms of components. Example Application.

We developed a pair of applications to help describe how Android applications operate. Interested readers can download the source code from our web sitepttp://siis.cse.psu.edu/android\_sec\_tutorial.html).

Let’s consider a location-sensitive social networking application for mobile phones in which users can discover their friends’locations. We split the functionality into two applications: one for tracking friends and one for viewing them. As Figure 1 shows, the FriendTracker application consists of components specifc to tracking friend locations (for example, via a Web service), storing geographic coordinates, and sharing those coordinates with other applications. The user then uses the FriendViewer application to retrieve the stored geographic coordinates and view friends on a map.Both applications contain multiple components for performing their respective tasks; the components themselves are classifed by their component types. An Android developer chooses from predefned component types depending on the component’s purpose (such as interfacing with a user or storing data).Component TypesAndroid defnes four component types:Activity• components defne an application’s user interface. Typically, an application developer defnes one activity per “screen.” Activities start each other, possibly passing and returning values. Only one activity on the system has keyboard and ocessing focus at a time; all others are suspended.Service components perform background processing. When an activity needs to perform some operation that must continue after the user interface disappears (such as download a fle or play music), it commonly starts a service specifcally designed for that action. The developer can also use services as application-specifc daemons, possibly starting on boot. Services often define an interface for Remote Procedure Call (RPC) that other system components can use to send commands and retrieve data, as well as register callbacks. Content provider

• components store and share data using a relational database interface. Each content provider has an associated “authority” describing the content it contains. Other components use the authority name as a handle to perform SQL queries (such as SELECT, INSERT, or DELETE) to read and write content. Although content providers typically store values in database records, data retrieval is implementation-specifc—for example, fles are also shared through content provider interfaces.Broadcast receiver

• components act as mailboxes for messages from other applications. Commonly, application code broadcasts messages to an implicit destination. Broadcast receivers thus sub-scribe to such destinations to receive the messages sent to it. Application code can also address a broadcast receiver explicitly by including the namespace assigned to its containing application.

Figure 1 shows the FriendTrack-er and FriendViewer applications containing the diferent component types. The developer specifes components using a manifest fle (also used to defne policy as described later). There are no restrictions on the number of components an application defnes for each type, but as a convention, one component has the same name as the application. Frequently, this is an activity, as in the FriendViewer application. This activity usually indicates the primary activity that the system application launcher uses to start the user interface; however, the specifc activity cho-sen on launch is marked by meta information in the manifest. In the FriendTracker application, for example, the FriendTrackerControl activity is marked as the main user interface entry point.

In this case, we reserved the name “FriendTracker” for the service component performing the core application logic.The FriendTracker application contains each of the four component types. The FriendTracker service polls an external service to discover friends’ locations. In our example code, we generate locaFriendTracker application BootReceiver Broadcast receiver ActivityFriendTracker FriendProvider Content provider Service FriendTracker control FriendViewer application FriendReceiver Broadcast receiver Activity FriendTracker Activity FriendViewer Figure 1. Example Android application. The FriendTracker and FriendViewer applications consist of multiple components of different types, each of which provides a different set of functionalities. Activities provide a user interface, services execute background processing, content providers are data storage facilities, and broadcast receivers act as mailboxes for messages from other applications.tions randomly, but extending the component to interface with a Web service is straightforward. The FriendProvider content provider maintains the most recent geographic coordinates for friends, the FriendTrackerControl activity defnes a user interface for starting and stopping the tracking functionality, and the BootReceiver broadcast receiver obtains a notifcation from the system once it boots (the application uses this to utomatically start the FriendTracker service).The FriendViewer application bis primarily concerned with showing information about friends’ locations. The FriendViewer activity lists all friends and their geographic coordinates, and the FriendMap activity displays them on a map. The FriendReceiver broadcast receiver waits for messages that indicate the physical phone is near a particular friend and displays a message to the user upon such an event. Although we could have placed these components within the FriendTracker application, we created a separate application to demonstrate cross-application communication. dditionally, by separating the tracking and user interface logic, we can create alternative user interfaces with different displays and features—that is, many applications can reuse the logic performed in FriendTracker.Component Interaction The primary mechanism for component interaction is an intent, which is simply a message object containing a destination component address and data.

The Android API defnes methods that accept intents, and uses that information to start activities (startActivity(Intent)), start services (startService (Intent)), and broadcast messages (sendBroadcast(Intent)). The invocation of these methods tells the Android framework to begin executing code in the target application. This process of intercomponent communication is known as an action. Simply put, an intent object defnes the “intent” to perform an “action.”One of Android’s most powerful features is the fexibility allowed by its intent-addressing mechanism. Although developers can uniquely address a target component using its application’s namespace, they can also specify an implicit name.

In the latter case, the system determines the best component for an action by considering the set of installed applications and user choices. The implicit name is called an action string because it specifes the type of requested action—for example, if the “VIEW” action string is specifed in an intent with data felds pointing to an image fle, the system will direct the intent to the preferred image viewer. Developers also use action strings to broadcast a message to a group of broadcast receivers. On the receiving end, developers use an intent flter to subscribe to specifc action strings. Android includes additional destination resolution rules, but action strings with optional data types are the most common.Figure 2 shows the interaction between components in the FriendTracker and FriendViewer applications and with components in applications defned as part of the base Android distribution. In each case, one component initiates communication with another. For simplicity, we call this inter-component communication (ICC). In many ways, ICC is analogous to interprocess communication (IPC) in Unix-based systems. To the developer, ICC functions identically regardless of whether the target is in the same or diferent application, with the exception of the security rules defned later in this article.The available ICC actions depend on the target component.

Each component type supports interaction specifc to its type for example, when FriendViewer starts FriendMap, the FriendMap activity appears on the screen. Service components support start, stop, and bind actions, so the FriendTrackerControl activity, for instance, can start and stop the FriendTracker service that runs in the background. The bind action establishes a connection between components, allowing the initiator to execute RPCs defned by the service. In our example, FriendTracker binds to the location manager in the system server.

## 中文译文

深入理解Android系统安全机制

下一代开放操作系统的主流将不会在桌面上，但是将会出现在我们每天携带的手机上。这些开放性的环境将会带领这些新的应用可能集成这些已经存在的在线服务，当然随着日以具增的数据与服务在手机上的支持，手机上的安全缺陷也越发明显。下一代操作系统本质在于是否提供一个完整综合的安全平台。

由开放手机联盟（open Handset Alliance 谷歌领导）所开发的android 系统是一个被广泛看好的一个手机开源系统，该系统提供一个基本的操作系统，一个中间件应用层，一个java开发工具和一个系统应用收集器（collection of system applications ）。尽管android SDK自2007年就发布了，但是第一部android 手机却在2008年10月才诞生。自从这时起谷歌开起了自己的时代，T-Mobile的G1的制造商台湾 HTC估算G1的发货量在2008年底已经超过100万部。据业内人士预期该G1手机的销量将会在2009年继续保持。不久的将来其他许多手机供应商要计划支持这个系统。

一个围绕android庞大的开发者社区已经建立，同时很多新的产品和应用已经可以在android上使用。一个Android的主要卖点是它使开发人员无缝把在线服务扩展到手机。这方面最明显的例子是谷歌的紧密集成Gmail，日历和联系人Web应用程序通过该系统。用户只需提供一个android用户名和密码，其手机自动同步与谷歌的服务。其他厂商正在迅速适应自己的现有的即时通讯，社交网络和游戏服务。Android和许多企业寻找新途径来整合他们的自己已有的业务到android上。

传统的台式机和服务器的操作系统一直在努力进行安全功能的集成。这些个人和商业应用在单一平台的很出色，然而这一块业务一个手机平台上像android上不是很有用。它给了许多研究人员希望。Android没有停在为其他平台体用应用支持：应用的执行依赖于顶层JAVA中间件，这个中间件运行在嵌入式Linux 内核之上。所以开发人员要把他们的应用部署到Android必须使用其自定义的用户界面环境。

此外，android系统应用限制各应用相互调用API协作，并且对方为自己的用户应用进行身份验证。尽管这些应用有一定的安全特性，我们一些有经验的开发人员开发android应用人士透露，设计安全应用程序并不总是直线前进的。Android使用一个简单的许可标签分配模式限制访问的资源，但其他应用程序的原因必要性和便利，其设计师们增加了困惑对这个系统。本文试图对Android的安全的复杂性进行讲解，并注意一些可能的发展缺陷以及应用程序的安全。我们通过尝试得出一些经验教训，希望对未来的安全有用。

Android Application

Android应用程序框架对开发者来说是一个强制架构。它没有一个main（）函数功能或单一入口点执行，相反，开发人员必须在设计方面的应用组件。我们开发的应用对android的sdk的帮助的API。

Example Application。

我们开发了一个描述如何创建android的应用。有兴趣的读者可以去我们的站点下载让我们考虑一个基于位置的社交网络应用，其中手机用户可以通过本应用发现他们的朋友们位置。我们进行功能拆分，分成两个应用程序：一个用于跟踪查看朋友和常看他们。如图1所示，FriendTracker应用包括跟踪的组件specifc朋友的位置（例如，通过一个Web服务），储存地理坐标，并分享这些合作统筹与其他应用程序。然后用户使用友情查看器应用程序来检索地理坐标和储存在地图上查看朋友。

这两个应用程序包含的多个组件包括展示自己任务，他们组件是由他们组件类型所决定的。 。一个Android 开发者选择从根据不同的组件类型组件的目的决定 （如与一个用户或存储数据接口）。

例如Android应用程序。FriendTracker和FriendViewer应用由多个不同类型的组件，每个提供一个不同的组功能。Activity提供一个用户界面，Service执行后台处理，Content提供存储，Broadcast receiver接收机其他应用程序的信息。

Component Types

android系通定义了4种组件类型。

Activity

组件定义应用程序的用户界面。通常，应用程序开发者定义每一个活动“画面。”Activity可以自己开始，也可能通过传递和返回值。在一时间只有一个键盘的系统Activity可以进行处理，在这个时候所有其他的Activity都会被暂停。

Service

组件执行后台处理。当一个活动需要进行一些操作，在用户界面消失以后（如下载一个文件或播放音乐），它通常采取此种动作特殊设计的服务。开发人员还可以在系统启动使用特殊的守护进程，Service通常定义一个远程过程调用（RPC），其他系统组件可以用来传送接口命令和检索数据，以及注册一个回调函数。

Content

组件存储和共享数据 用关系数据库接口。每个Content供应者都有一个关联的“权限”来形容它的内容包含。其他组件使用时作为一个handle执行SQL查询（例如SELECT，INSERT或DELETE内容。虽然Content供应者通常存储把数值放在数据库记录中，数据检索是实现特殊的例子，文件也同时通过内容提供商共享接口。

Broadcast receiver

该组件作为为从邮件信箱发送信息给他应用程序。通常，广播消息的应用程序代码隐含的目的地。因此，广播接收器订阅这些目的地接收发送给它的消息。应用程序代码也可以解决明确广播接收机包括命名空间分配。

图1显示了FriendTracker和FriendViewer应用所包含的不同的组件类型。开发者组件使用一个主要定义文件（也用于定义权限，稍后介绍）。上有一个应用程序的组件的数量没有限制定义每种类型，但作为习惯，一组件应具有相同的名称该应用程序。通常情况下，这是作为在FriendViewer activity中进行注册。这一动作通常指示主activity作为该系统应用程序启动器用于启动用户界面;然而，如果需要启动特定的activity，开发者需要者在选择配置manifest 信息来实现这一个功能。在在FriendTracker应用，例如，FriendTrackerControl活动被标记为主用户界面的启动点。在这种情况下，我们保留名称“FriendTracker”为服务执行的核心组成部分的应用程序逻辑。

在FriendTracker应用包含四种类型的组件。在FriendTracker服务搜寻调查外部服务并发现好友的位置。在我们的示例代码中，位置是我们随机生成的，但直接通过网络连接组件接口的服务。该FriendProvider Content 提供保持最新的朋友地理坐标，FriendTrackerControl活动用于启动和用户界面停止跟踪好友功能，该系统一旦启动 BootReceiver通知从广播系统启动。

该FriendViewer应用主要是显示有关好友的位置的信息。每个启动的FriendViewer将会列出了所有的朋友和他们的地理坐标，FriendMap显示他们在地图上的位置。FriendReceiver将会等待接收附近的手机发送的消息这个消息来一个所指定的朋友。尽管我们可以在这些组件内放置在FriendTracker应用，但是我们仍然创建了一个单独的应用程序来展示跨应用的沟通。此外，通过分离程序功能和接口，我们可以创建不同的显示和功能，可选用户界面是，许多应用程序可以重用这些来自FriendTracker的功能。

Component Interaction

该组件交互的主要机制是一个intent ，这是一个简单的消息对象，其中包含一个目的地组件的地址和数据。 Android的API定义了他的方法中传入intent ，并使用该信息来启动一个activity例如开始一个activity(startActivity(intent))，启动服务（startService（intent））和广播信息（sendBroadcast（intent））。Android框架来通知这些方法的调用开始执行在目标应用程序代码。这个过程中内部组件通信称为一个动作。简单地说， Intent对象定义的“Intent”以执行“action”。Android的一个最强大的特点是允许的多种intent寻址机制。开发人员可以解决一个目标组件使用其应用的空间，他们也可以指定一个隐含的名称。在后一种情况下，系统决定了一个action的最佳组件，通过考虑安装的应用程序和用户的选择 。

这个隐含的名字被称为动作字符串因为他特殊的类型的请求动作。例如一个view动作字符串，在一个intent中和数据域指向一个图像文件，系统将会直接指首选图像浏览器。

开发者也能使用动作字符串进行大量广播发送和接收。在接收端的接收者，开发者使用一intent 过滤器来定制特殊的动作字符串。Android系包括附加目标的决议规则，但可选的数据操作字符串类型是最常见的。

显示了组件之间的FriendTracker和FriendViewer应用程序和组件的交互作用在应用程序中定义为基础的Android发布的一部分。在每一种情况下，发起一个组件与其他的沟通。为了简单起见，我们称这个为件间通信（ICC）。在许多方面，ICC是类似于进程间通信（IPC）在基于Unix的系统中。对于开发人员，ICC的功能相同无论目标是在相同或不同的应用与界定将在下文的安全规则中说明。

可用的ICC的动作取决于目标的组成部分。每个组件类型支持自己的类型。例如，当FriendViewer开始FriendMap的FriendMap活动出现在屏幕上。服务组件支持启动，停止，并结合行动，所以FriendTrackerControl活动，例如，可以启动和停止FriendTracker服务在后台运行。Action的绑定组件之间建立连接，使启动执行的服务定义的RPC。在我们的例子，FriendTracker结合到系统中的服务器位置的管理。

# 致 谢

在本次毕业设计中，我得到了指导老师刘兵的悉心指导与帮助。刘老师对工作一丝不苟的态度让我很感动。无论在理论上还是在实践中，都给予了我很大的帮助。

同时也感谢亚星联创公司的同事和项目leader对我的帮助。感谢你们的温切关怀和公司的温馨办公环境。校外实习培训期间，感谢管经理的全力协助下，也许是受公司良好的环境氛围影响，这次系统的界面色调来自于公司的环境色调。

在论文即将完成之际，我的心情无法平静，从开始进入课题到论文的顺利完成，有多少可敬的师长、同学、同事、朋友给了我无言的帮助，在这里请接受我诚挚的谢意！