北京交通大学

硕士学位论文

# 基于Android平台即时通信系统的设计与实现

姓名：马志强

申请学位级别：硕士

专业：软件工程

指导教师：李伟生

20090601

# 中文摘要

摘要：即时通信系统产生以来，这种以网络为基础的、与其它在线用户进行信息交互的实时方式，以其方便快捷的特点，受到了对消息反馈即时性要求很高的商业和服务行业的青睐。传统的即时通信应用大部分是在PC端实现的，但随着无线传输网络的发展，各种智能操作平台的推出提供了硬件基础，在移动设备上的即时通信系统的研究成为了3G业务新的亮点。本文为解决移动设备与PC端即时通信以及通信协议缺乏统一标准的问题，分析了当前的研究现状，设计并实现一个Android平台上使用Jabber协议的移动即时通信系统。

本文采用软件工程的管理方法，对项目的需求进行了分析，完成了功能用例建模，从HTTP传输、Jabber协议解析到客户端的各个模块部分进行分析设计，提出了本系统的体系结构和整体架构设计方案，阐述了系统实现应用的关键技术，建立了可扩展的会话模型，采用松散耦合的方式设计、实现了Android平台上基于Jabber协议的移动即时通信系统。

经过测试运行，结果表明，系统完成的用户注册登录、好友管理、分组管理以及信息交互等即时通信功能能够顺畅运行，解决了与不同通信协议对接进行交互的问题，获得了良好的用户体验，已用于实际工程项目中。

关键词：Android；Jabber；即时通信系统；移动设备应用；3G；松散耦合；会话模型

分类号：TP311.52

# ABSTRACT

**ABSTRACT:** Since instant messaging systems appeared, such web-based real-time way which other online users to interact with, is favored by demanding real-time business and service industries with its convenient features. Traditional real-time communications applications are mostly realized in the PC side, but with the development of wireless transmission networks, and the introduction of intelligent operating platform provides a hardware foundation, mobile devices in real-time communications system has become the new 3G business highlights. In this paper, to solve the interaction between mobile device and PC instant messaging client, as well as a lack of unified communication protocol standards, analysis of the current status of research, design and realize an instant messaging system of Android platform-based using Jabber protocol.

In this paper, adopt the management methods of software engineering, analyze the project requirements, complete the functional use case modeling, from the HTTP transmission, Jabber protocol analysis to each module of client to analyze, design of the proposed architecture of the system and the overall structure design, introduce key technologies of the application, establish a scalable model of conversation, use design methods of loosely coupled, realize instant messaging system of Android platform-based using Jabber protocol .

After the test of the application, results show that the completion of the user registration system, registry, friends management, group management and information exchange, such instant messaging features have run smoothly, solved the problem of interaction among different instant messaging protocols , get a good user experience, the system has been used in actual projects.

**KEYWORDS:** Android; Jabber; Instant Messaging System; Mobile Application; 3G; Loosely Coupled; Conversation Model.

**CLASSNO:** TP311.5

# 独创性声明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作和取得的研究成果，除了文中特别加以标注和致谢之处外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京交通大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

学位论文作者签名：马志强 签字日期：2009 年 6 月 30 日

# 学位论文版权使用授权书

本学位论文作者完全了解北京交通大学有关保留、使用学位论文的规定。特授权北京交通大学可以将学位论文的全部或部分内容编入有关数据库进行检索，提供阅览服务,并采用影印、缩印或扫描等复制手段保存、汇编以供查阅和借阅。同意学校向国家有关部门或机构送交论文的复印件和磁盘。

（保密的学位论文在解密后适用本授权说明）

学位论文作者签名：马志强　　导师签名：李伟光

签字日期：2009年6月30日　　签字日期：2009年6月30日

# 致谢

本论文的工作是在我的导师李伟生教授的悉心指导下完成的，李伟生教授严谨的治学态度和科学的工作方法给了我极大的帮助和影响。在此衷心感谢三年来李伟生老师对我的关心和指导。

李伟生教授悉心指导我们完成了实验室的科研工作，在学习上和生活上都给予了我很大的关心和帮助，在此向李伟生老师表示衷心的谢意。

李伟生教授对于我的科研工作和论文都提出了许多的宝贵意见，在此表示衷心的感谢。

在实验室工作及撰写论文期间，孙光明、包尔固德等同学对我论文中的研究工作给予了热情帮助，在此向他们表达我的感激之情。

另外也感谢家人，他们的理解和支持使我能够在学校专心完成我的学业。

# 1 绪论

## 1.1 项目背景

即时通信（Instant Messaging, IM）是随着互联网的出现而兴起的新型通信手段，根据对通讯软件发展现状的分析与研究[1]，其发展十分迅速，受到了广泛地关注。本项目来自于以下当前最受关注相关应用的结合点：

1. 开放性移动设备Android平台目前已受到了很多人的关注，使用该平台的手机也已经推出，并获得广泛的好评，基于此平台应用程序的开发也逐渐成为一个热门的方向；

2. 固定网络的即时通信系统在网络中的广泛应用，用户已经将即时通信系统作为生活中不可或缺的通信工具；

3. 移动网络中的无线终端设备应用日益广泛。在中国手机的普及率已经很高，通过便携的、无线的移动设备访问Internet已经成为人们需求的方向，用户迫切希望在手机、PDA等便捷的无线终端设备上与桌面即时通信系统进行交互[2]。

移动终端平台中实现即时通信系统作为移动通信业务的扩展，将对移动业务产生重要影响，也必将获得更为广大的市场价值。08 年初，播思通讯公司与中国移动合作，欲推出基于Android的OMS（Open Mobile System）系统，来对抗近期流行的iPhone，以在智能手机市场上分到一杯羹。经过半年的努力，OMS系统已经基本成形，播思通讯公司已经完成了Android平台的本地化工作，并根据中国移动公司的需求内置了一些手机操作软件，比如手机输入法、音乐播放器、手机邮箱等。本系统是公司在Android平台Fetion已经完成的基础上，通过对目前市场上应用的多个即时消息协议和无线搭建平台的对比分析，立项以构建一个Android平台上基于Jabber协议框架的移动即时通信系统。

## 1.2 研究的目的和意义

即时通信软件作为一种便捷的网络通信技术已经越来越深入人心，应用范围从单纯的网络聊天工具变成工作生活所不可缺的信息交流平台。在互联网日益普及的今天，即时通信的用户规模也呈现出快速增长的态势。

现阶段，用手机等移动设备和即时通信软件挂钩，把以往的只能应用在 PC 机

上的即时通信软件移植到移动设备中，让用户能够更方便地应用即时通信产品，是即时通信的发展趋势，也是IM系统软件市场发展的一个重要方向。移动即时通信系统实现主要来自最近通信界最成功的两个应用的结合点：固定网络中的桌面即时通信和移动网络中的短消息系统。移动通信的即时通信服务就是在传统的基于Web通信系统的概念上，把手机的短信和手机移动互联网完美地结合起来，使用户通过移动设备终端，也能够方便地与他人以短信、移动互联网来进行实时的信息交流，它突破了传统Web界限，把即时信息转移到移动互联网上面，同时用户通过短消息或移动互联网，实现更即时的交流。可以想象，如果用户在移动终端上能够像桌面即时通信一样方便地使用并且可以访问已有的朋友列表，无线即时通信系统就能够不费力地获得极大的市场份额。

长期以来，各个IM软件厂商推出的IM软件相互独立，缺乏统一标准，使得各种IM系统之间互相发送信息较为困难，例如AOL与Yahoo，MSN与AOL，MSN与QQ等，他们之间相互通信的难度较大。因此，在这个时候，一个统一的协议就显得格外重要，这种统一协议应当可用于E-mail、Web和语音流的简单邮件传输协议（SMTP）、HTTP和实时协议（RTP）等IM应用。由Jabber组织发展的Jabber协议，是包含支持符合IETF规范的即时消息协议和Presence技术的基本协议，由此协议形成了XMPP，XMPP（Extensible Messaging and Presence Protocol，可扩展消息处理现场协议）为用于现场的消息路由处理的XML数据流协议，在即时通信到场协议中得到广泛的应用[3]，并被确立为IETF标准。XMPP能够提供最佳路由优化处理方案[4]，保证大量敏感数据在复杂的交互节点中即时传输。它是即时消息处理（Jabber IM）系统的基准协议技术，能为多网络间连接提供安全和易于实现的编程语言环境。本文的研究目的就是根据对Jabber和XMPP的研究[5]，设计和实现一个在移动平台上，能够兼容多种即时通信协议的IM系统。

本文介绍了在移动平台上的即时通信系统的开发，作为Google最新推出的移动设备平台，Android平台与其它Symbian、Windows Mobile手机操作系统相比，其源代码完全开放，任何人和机构都可以免费使用，Android研发已经逐渐成为了一个热门的方向，而其他操作系统的吸引力正在下降。中国移动旗下的播思通讯公司正在研究基于Android的操作系统，并把业务层应用替换成自己的应用，命名为OMS（Open Mobile System）系统，并内置了必要的系统软件。鉴于中国移动在国内的影响力，OMS系统必然会在将来的操作系统大战中占据一席之地。由于播思通信公司与中国移动的合作关系，在OMS内置了飞信（Fetion，中国移动通信的即时通信系统，由于与手机的捆绑关系，目前也有很广阔的用户群）客户端，是基于Fetion协议的即时通信系统。

Jabber即时通信协议是基于XML流的协议和技术，采用Jabber协议的框架可

以比较容易地实现一个即时通信系统，Jabber 协议相对于目前比较流行的商业即时通信系统，开放性是其最大的优点，并提供了多种服务器和客户端系统、组件以及代码类供开发人员进行详细研究。Jabber 是基于开放源代码的，用户可以免费地得到，Jabber 不在任何组织或厂商的商用控制中，任何人或组织都可以为了满足自己的需求来定制服务，并且 Jabber 是基于 XML 的协议，易于实现以及扩展[6]。在此，本文提出了在 Android 平台上基于 Jabber 协议的即时通信系统的设计和实现，可以使得任何基于 Jabber 协议的 IM 应用均可以在此系统中使用。

## 1.3 国内外研究现状

即时通信软件的发展突飞猛进，即时通信所拥有的实时性、跨平台性、成本低、效率高等诸多优势，使之成为网民们最喜爱的网络沟通手段之一。

根据赛迪发布的 IM 市场发展综述[7]，2008 年中国企业即时通讯市场终端用户规模达到 2236 万人，与 2007 年的 1813 万人相比，同比增长 23.3%，具体见图 1.1。由于受到全球范围金融危机影响，企业即时通讯市场增速有所放缓，但是市场的潜在需求仍然没有全部释放，一旦经济回暖，市场将延续前期快速增长的趋势。

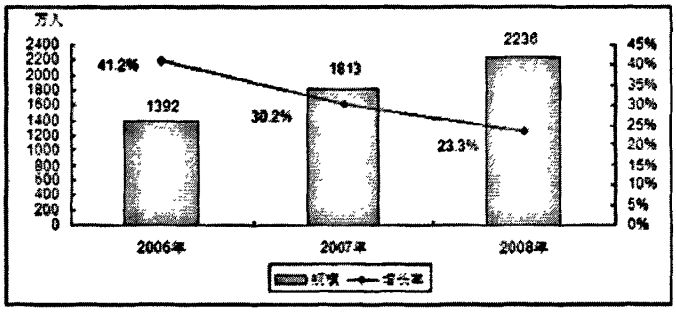


图1.1 2006－2008年中国企业即时通讯市场用户规模增长

Figure 1.1 The scale of growth in 2006-2008 IM market in Chinese enterprises

而对于个人即时通信系统市场，国内来说，自 QQ 问世以来，以 70%的市场份额占据国内 IM 软件市场的第一把交椅，遥遥领先其它的对手，成为最流行的即时通信软件，为用户提供各种服务；国际上，MSN 是微软公司推出的即时通信产品，也是目前全球最流行的聊天工具之一，它以强大的后台以及操作简单而备受广大网民的喜爱。而且精明的微软管理层，把 MSN 与 Windows 操作系统捆绑，国人便很快对 MSN 家喻户晓了。与 QQ 过于偏重休闲娱乐不同，微软 MSN 更偏重于商务应用，使之成为企业职工们相互通信的首选工具。事实证明，这种定位

是正确的，在 QQ 已经占据国内的即时通信市场的绝大多数市场份额的前提下，微软能够成功地从 QQ 手里抢到相当一大部分市场份额。除了微软的 MSN 之外，比较成功的还有 AOL、雅虎通、Skype，GTalk 等。其中 Skype 是互联网语音沟通工具，采用了最先进的 P2P 技术，为用户提供了超清晰的语音通话效果，使用端对端的加密技术以保证通信的安全可靠。Google 的 GTalk 是一款与 Gmail 集成的即时通信工具，并遵从 jabber 协议，且 Google 已经将 Gtalk 的服务器开放给了其它的 Jabber 服务器。

08 至 09 年国内外即时通信系统市场处于导入期，金融危机将促进市场竞争格局的形成，高效、稳定、安全成为服务的重点。信息化建设的加速将促进企业即时通信市场的发展，统一通信将成为发展的趋势，未来的即时通信服务商能够提供短信、邮件、电话等多渠道的解决方案，支持多媒体服务平台，为用户提供更为方便的应用，把即时通信向移动设备方向上转换。

当前移动设备的市场上主要有 Android、Windows Mobile、Symbian 和 iPhone 等开放手机平台，用户将获得更多的移动应用，而移动终端业务开发也会蓬勃兴起。关于各种平台的比较如下：

**1. Symbian**

目前包括 Nokia、三星在内的大型终端厂商都有大量采用 Symbian 平台的产品，同时也拥有这方面的丰富设备产品和应用产品的经验。用户层面上，目前使用 Symbian 平台手机的用户也是最多的。但以 Nokia 为代表的 Symbian 队伍有两个弱点，一是开源不彻底；二是目前中国用户手机占有率最高的 Nokia 摇身一变成为了“运营商”，通过手机预置连接 Nokia 门户网站的业务程序来提供增值业务，而且拒绝中国本地运营商的业务预置。相对而言，Android 的免费对中小型手机厂商吸引力还是很大的，而且运营商也希望通过开源免费的 Android 定制自己的操作系统以达到控制终端业务的目的。

**2. Window Mobile**

微软的 Windows Mobile 不会开源的，还是向手机厂商收着软件版权费。WM 终端的市场占有率不是太高，但是推出一段时间来，也吸引了不少高端手机采用。跟 Symbian 的理由一样，相信未来中小型的厂商还是会趋向采用免费的 Android。

**3. iPhone**

iPhone 手机的商业模式跟上面两种都不太一样，这决定于 iPhone 手机硬件与软件一一绑定的特性，Apple 采用与运营商分成的策略。在美国来看，运营商在 iPhone 预植业务是可以的，但是大部分的业务获得都是通过 Apple 的门户提供的，中国运营商不太愿意引入。iPhone 虽然开放了 SDK，可是却不能在 PC 机上使用，这对于中国的应用开发市场是个不小的阻碍。

总得来说，Symbian 太强势、Windows Mobile 太贵、iPhone 太封闭都是相比 Android 之下的弱点。

从中国范围来看，中国移动走在 Android 潮流的前列，已经完成了 Android 的定制，还开发了不少自有的中间件和应用，为以后的业务预置铺垫了基础。相信未来 Android 将在移动终端平台市场占有一席之地，在 Android 平台上的应用也会受到更多的关注。

## 1.4 论文的主要内容和组织结构

本文的研究工作是设计和实现一个 Android 平台的即时通信系统，实现各种类型客户端之间的互操作性功能，提供好友的实时状态，实现移动设备与 PC 终端的互通，使用户能通过手机等移动设备的即时通信系统客户端[8]能够随时随地与他人进行即时消息通信。

本系统采用 Jabber 协议，在 Google 最新推出的 Android 平台下进行研究工作。本文的研究内容主要有下面几方面：

1. Android 平台上即时通信系统的架构：提出整个系统的合理架构以实现整个系统；

2. Android 开发平台：介绍 Android 平台的相关知识，深入阐述如何在 Android 平台下进行程序开发；

3. Jabber 协议：介绍 Jabber 协议的内容并对协议进行解析；

4. 移动即时通信系统的实现：根据提出的系统架构，并阐述本移动即时通信系统的设计和具体实现，最终对系统进行测试工作。

本文在第二章介绍项目所用到的基本知识，对 Android 平台以及 Jabber 协议进行介绍，讨论在 Android 平台下程序的开发以及 Jabber 协议的主要内容、结构。

然后在第三章根据要实现的即时通信系统的特点，对市场上类似的产品进行详细分析，对比其中的差别，来明确本系统的可行性，提出系统的需求，对系统进行总体分析。

确定系统需求后，如何建立一个相对合理的系统架构和对系统进行实现，是本文的讨论重点，这部分在第四章。在系统详细设计中，对整个系统的服务器设计、客户端设计、数据持久化设计以及会话模型进行详细介绍，然后在第五章详细介绍系统实现，对客户端结构中的各个模块实现进行阐述。

最后第六章中对本文系统的实现功能以及运行状态进行总结。

# 2 系统相关协议分析与基础技术介绍

## 2.1 Android 平台

Android 平台是 Google 于 2007 年 11 月推出的一种智能手机平台，它是由一个由操作系统、中间件、用户友好界面和应用软件组成，全面整合的软件栈。Android SDK 提供了在 Android 平台上使用 Java 语言进行 Android 应用开发必须的工具和 API 接口。

### 2.1.1 Android 平台的组成

Android 不仅是一种操作系统，它更是一个开源的体系架构。Android 平台大量应用了开源社区的成果，并将其针对移动设备进行了一系列地优化。Android 平台包含的主要构成部分及其特性有：

- 经过 Google 改进和调优的 Linux Kernel；
- 经过 Google 修改的 Java 虚拟机 Dalvik VM；
- 大量可用的类库和应用软件，例如浏览器 WebKit，数据库 SQLite；
- Google 已经开发好的大量现成的应用软件，并可以直接使用很多 Google 在线服务；
- 基于 Eclipse 的完整开发环境；
- 优化过的 2D 和 3D 图形处理系统；
- 多媒体方面对常见的音频、视频和图片格式提供支持；
- 支持 GSM，蓝牙，EDGE，3G，WiFi，摄像头，GPS。

### 2.1.2 Android 应用程序框架

Android 平台的架构[9]从上至下包含了五个部分：应用程序（Applications）、应用框架（Application Framework）、开发库（Libraries）、运行时环境（Android Runtime）以及 Linux 内核（Linux Kernel），如图 2.1 所示：

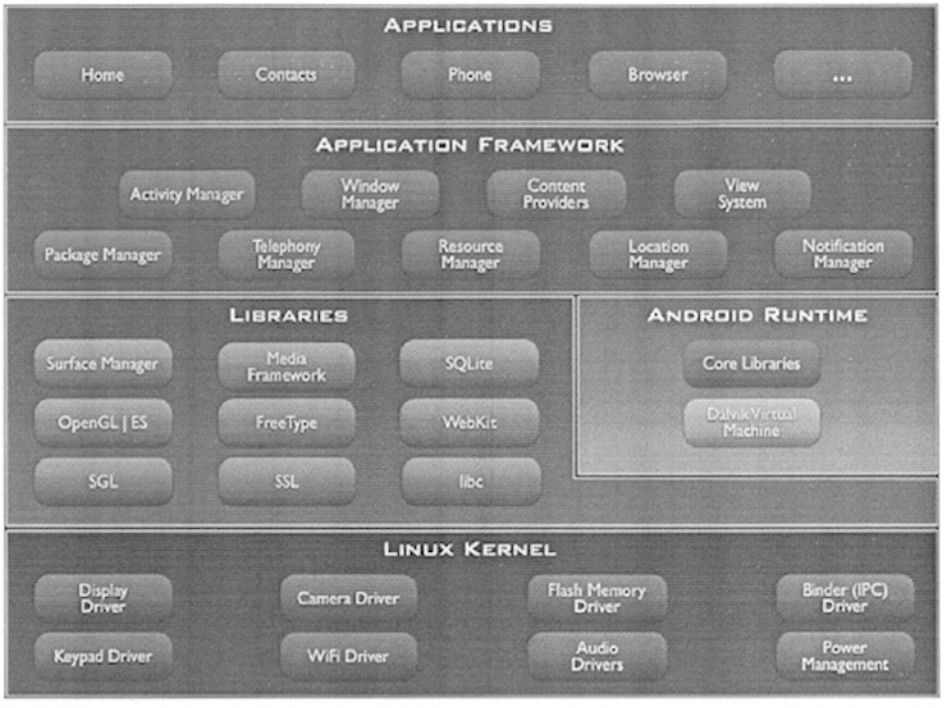


图 2.1 Android 应用程序框架图

Figure 2.1 Android Application Framework Chart

- **应用程序（Applications）**

Android 会同一系列核心应用程序包一起发布，该应用程序包包括 email 客户端，SMS 短消息程序，日历，地图，浏览器，联系人管理程序等。所有的应用程序都是使用 Java 语言编写的。

- **应用程序框架（Application Framework）**

开发人员也可以访问核心应用程序所使用的 API 框架。该应用程序的架构设计简化了组件的重用，任何一个应用程序都可以发布它的功能块并且任何其它的应用程序都可以使用其所发布的功能块（不过得遵循框架的安全性限制）。同样，该应用程序重用机制也使用户可以方便的替换程序组件。

隐藏在每个应用后面的是一系列的服务和系统，其中包括：

1．丰富而又可扩展的视图（Views），可以用来构建应用程序， 它包括列表（lists），网格（grids），文本框（text boxes），按钮（buttons）， 甚至可嵌入的 web 浏览器。

2．内容提供器（Content Providers）使得应用程序可以访问另一个应用程序的数据（如联系人数据库）， 或者共享它们自己的数据。

3．资源管理器（Resource Manager）提供非代码资源的访问，如本地字符串，

图形，和布局文件（layout files）。

4. 通知管理器（Notification Manager）使得应用程序可以在状态栏中显示自定义的提示信息。

5. 活动管理器（Activity Manager）用来管理应用程序生命周期并提供常用的导航回退功能。

- **开发库（Libraries）**

Android 包含一套 C/C++开发库，主要包括：libc、Media Framework、WebKit、SGL、OpenGL ES、FreeType、SQLite 等。它们被应用于 Android 系统的各种组件中。这些功能通过 Android 应用框架展现给开发人员。

- **运行时环境（Android Runtime）**

Android 包括了一个核心库，该核心库提供了 JAVA 编程语言核心库的大多数功能。每一个 Android 应用程序都在它自己的进程中运行，都拥有一个独立的 Dalvik 虚拟机实例。Dalvik 被设计成一个设备可以同时高效地运行多个虚拟系统。Dalvik 虚拟机执行（.dex）的 Dalvik 可执行文件，该格式文件针对小内存使用做了优化。同时虚拟机是基于寄存器的，所有的类都经由 JAVA 编译器编译，然后通过 SDK 中的"dx"工具转化成.dex 格式由虚拟机执行。Dalvik 虚拟机依赖于 linux 内核的一些功能，比如线程机制和底层内存管理机制。

- **Linux 内核（Linux Kernel）**

Android 的核心系统服务依赖于 Linux 2.6内核，如安全性，内存管理，进程管理，网络协议栈和驱动模型。Linux 内核也同时作为硬件和软件栈之间的抽象层。

整体架构上来看，Android 相比其它平台显示出了自身的特点，如集成了 WebKit 浏览器、Dalvik 虚拟机等模块。

### 2.1.3 Android 应用程序类型分析

Android 的 API 主要包含几个部分：Views、Intents、Activity、Permissions、Resource Type、Services、Notifications、Content Providers 以及 XML 支持。比较重要且常用的如 Views 用于提供界面设计的接口；Services 提供运行在后台的服务；Content Provider 定义了一组系统级的数据库；Notification 为用户提供提醒的 API 等。这些都可以通过 SDK 附带的文档[10]查询到。

Android 上的应用程序可以分成四种主要类型：活动、服务、接收器和 Content Provider。

- **活动（Activity）**

活动是最常用的 Android 应用程序形式。活动在一个称为视图类的帮助下，为

应用程序提供 UI。视图类实现各种 UI 元素，如文本框、标签、按钮等。一个应用程序可以包含一个或多个活动。这些活动通常与应用程序中的屏幕形成一对一的关系。

应用程序通过 startActivity()或 startSubActivity()方法从一个活动转到另一个活动。如果应用程序只需切换到新的活动，使用前一个方法；如果需要异步的调用/响应模式，就使用后一个方法。但均需要通过方法参数传递一个 Intent，由操作系统负责决定哪个活动最适合满足指定的 Intent。

Intent 是 Google 在 Android 体系结构中引入的一种新颖的设计元素，Intent 是一种构造，应用程序可以通过它发出相应的请求。应用程序可以按照相似或互补的方式进行注册 IntentFilter，表明他们有能力或有兴趣执行各种请求或 Intent。

- **视图（View）**

Android 活动通过视图显示 UI 元素。视图采用以下布局设计之一：LinearVertical，LinearHorizontal，Relative，Table。选择一种布局之后．就可以用各个视图显示 UI。视图元素由一些 UI 元素组成，包括：Button，EditText，CheckBox，RadioButton，List，Grid，DatePicker，TimePicker 等。视图是在一个 XML 文件中定义的。每个元素有一个或多个属于 Android 命名空间的属性。

- **服务（Service）和接收器（Reciever）**

与其他多任务计算环境一样．Android 可以在后台运行着一些应用程序。Android 把这种应用程序称为服务。服务是没有 UI 的 Android 应用程序。

接收器是一个应用程序组件，它接收请求并处理 Intent。与服务一样。接收器在一般情况下也没有 UI 元素。服务和接收器通常都在 AndroidManifest.xml 文件中注册。

- **通过 Content Provider 进行数据管理**

Content Provider 是 Android 的数据存储抽象机制。Content Provider 对数据存储的访问方法进行抽象，在许多方面起到数据库服务器的作用。对数据存储中数据的读写操作应该通过适当的 Content Provider 传递．而不是直接访问文件或数据库。

## 2.2 Jabber 协议

### 2.2.1 Jabber 协议简介

Jabber 是一个由开源社区发起并领导开发的即时消息和在线状态的系统。Jabber 系统和其它即时消息服务的一个功能上的差别在于 Jabber 拥有开放的 XML

协议。采用 Jabber 提供的协议框架可以很容易地实现一个即时通信系统，满足人们目前和将来对即时通信的需求。相对于目前流行的商业即时通信系统（如 MSN、QQ 等），Jabber 是完全开放的协议和标准[11]，允许任何人使用 Jabber 协议构建系统，并提供多种服务器和客户端系统、组件、代码类供下载研究和使用，所有的一切均为开放源代码的，用户可以免费、自由地得到。Jabber 不在任何组织和商用厂商的控制之下，任何人都可以为满足自己的需求定制 Jabber 服务。Jabber 开源社区为了解、使用 Jabber 协议提供了一个交流的渠道。

Jabber 协议充分利用了 XML 技术的各种优点，不但人机都可以对通信发送的消息进行识别，易于对系统进行设计和实现，而且很容易扩展现有协议来实现新的功能。

Jabber 网络是基于服务器的（即客户端之间彼此不直接交谈），也是分布式的。相比 AOL 即时通或 MSN Messenger 等服务，Jabber 没有中央官方服务器。Jabber.org 的公众服务器上有大量的用户，事实上任何人都可以在自己的网域上运行 Jabber 服务器。

Jabber 识别符（JID）是用户登入时所使用的帐号，看起来通常像一个电子邮件地址，如 someone@example.com；前半部分为用户名，后半部分为 Jabber 服务器域名，两个字段以@符号区隔。

### 2.2.2　Jabber 网络模型

Jabber 使用的是客户端——服务端的系统架构，而不是其它一些即时消息系统使用的客户端——客户端系统架构。所以从一个客户端发给另一个客户端的 Jabber 消息和数据都必须通过服务器，并可以通过服务器之间的数据交换使得与不同协议之间进行信息交换[12]。

Jabber 系统包括客户端和服务器端，在 Jabber 里，客户端叫做节点，服务器端分三部分：主机（host），即我们所说的服务器、服务端（service）和网关（gateway）。

Jabber 网络采用消息网关与其它类型的消息网络进行交换，具有和其它消息系统的最广泛协同操作性。任何 Jabber 网络既可以连入公共 Jabber 网络，也可以单独孤立存在。

Jabber 的工作原理见图 2.2：

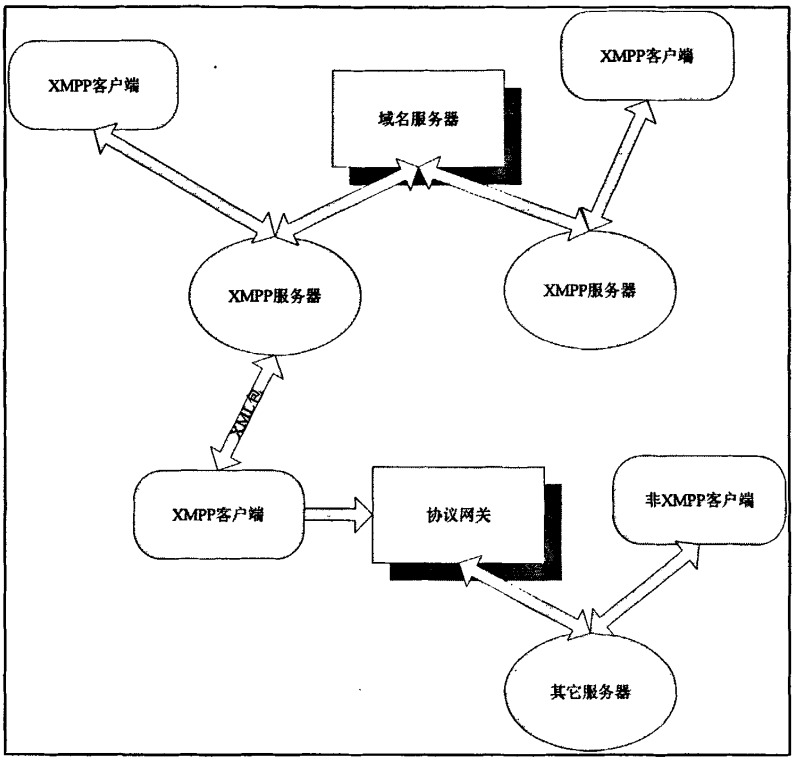


图 2.2 Jabber 工作原理图

Figure 2.2 Jabber Working Principle Diagram

Jabber 服务的工作步骤为：

1.节点连接到服务器；

2.服务器利用本地目录系统中的证书对其进行认证；

3.节点指定目标地址，让服务器告知目标状态；

4.服务器查找、连接并进行相互认证；

5.节点之间进行交互。

### 2.2.3 Jabber 会话模型

Jabber实体之间相互传送数据使用XML流。在两个实体的连接期间，XML流从一个实体流向另一个实体，例如客户机与服务器之间的会话由两个平行的XML流组成。当一个Jabber客户端连接上一个Jabber服务器时，这个客户端将发起一个客户端到服务器的XML流，同时作为响应，服务器也将发起一个服务器到客户端的XML流。

由于会话流中不同协议消息通过不同的XML文档元素组成，因此Jabber协议为防止内部协议之间相互冲突引入了命名空间（Namespace），为协议中的文档元素

提供一个上下文环境。也就是说不同的协议只要通过不同的命名空间加以区分可以使用相同的元素名称或者属性名称。命名空间（Namespace）的名称是统一资源标识符（Uniform Resource Identifier,URI），以HTTP URL（统一资源定位器）或者URN（统一资源名称）格式进行书写。Jabber通过URN格式已经定义了很多命名空间，来规范目前的标准协议。例如“jabber:iq:roster”命名空间确定了好友花名册管理协议，“jabber:iq:auth”命名空间确定了简单用户认证协议等等。

XML流通过初始会话元素（<stream/>）进行控制。双方会话都在<stream/>文档元素提供的上下文环境中，通过发送元素的开始标签<stream:stream>开始双方的会话，以元素结束标签</stream:stream>的发送结束双方的会话。其中以“stream”为别名的命名空间"http://etherx.jabber.org/streams"对初始会话元素表示的协议进行规范，另外默认命名空间“jabber:client”整个会话提供上下文环境。在XML会话流中包括了<message/>, <presence/>和<iq/>三个顶级的XML元素，以下将对这些元素分别描述。

即时消息系统最基本的功能就是能够在两个用户之间实时的交换消息，即时消息元素<message/>就提供了这个功能，其中包括了多个属性和子元素。最重要的两个子元素是<subject/>和<body/>，分别表示消息主题和消息正文。属性“from”和“to”分别表示了消息发送者和接收者的地址。可选的“type”属性给接收者一个提示，通知该消息所属的类型。

存在状态元素<presence/>用来传递用户存在状态的感知信息。用户的连接状态可能是“可用的(available)”或“不可用的(unavailable)”，用户的连接状态通过“type”属性进行设置，其中“available”为默认值。另外type属性还可以用于联系人状态订阅的设置，例如“subscribe”值表示向某一联系人请求订阅状态，“unsubscribe”值表示向某一联系人取消订阅状态。当用户的状态是可用时，好友发给他的消息就会立即被传递，否则好友发给他的消息就会被服务器存储起来，直到用户状态变为可用时。用户在连接服务器时，能够自己控制存在的可用性。服务器会将用户状态的改变即时的通知给所有订阅了该用户存在信息的所有用户。

请求响应元素<iq/>用来发送和获取实体之间的信息。该元素类型的消息是通过“请求/响应”机制在实体间进行交换，这和HTTP中的“GET"和“POST”方法相类似。元素通过“type”属性设置状态。id属性对请求响应的结果进行跟踪。由于<iq/>元素用于不同目的，他们之间通过<query/>子元素中所指的命名空间进行区别。<iq/>元素通过“请求/响应机制”的应用很容易设计新的协议。

Jabber网络通信是基于异步的消息模式，当通信发送者向接收者发送消息时，前者不需要等待后者处理完消息即可继续执行下去，后者可以对消息进行缓存并选择在合适时间进行处理。本文在这里只介绍客户机和服务器之间的通信，服务

器之间的通信类似。基于Jabber协议的通信[13]可以简单描述如下:

1．Jabber客户端通过5222端口连接Jabber服务器;

2．客户端向服务器发送<stream/>元素的开始标签，其中包含服务器相关信息:

3．客户端等待服务器应答<stream/>元素的开始标签，其中包含一个唯一的会话ID;

4．客户端通过约定的安全鉴定协议(<iq/>元素)以合法的用户身份登录系统;

5．客户机和服务器之间互相发送<presence/>,< message/>或<iq/>元素格式的通信信息;

6．通信一方发送</stream>元素的结束标签，结束Jabber通信会话;

7．双方关闭网络连接。

# 3 系统需求分析

## 3.1 可行性分析

当前移动设备即时通信市场上流行的 IM 客户端主要有 QQ、MSN 等，在 Windows Mobile、Symbian 60 等平台中均有相关实现，但 Android 平台是 Google 最新推出的移动设备操作系统，在此领域还没有相关的即时通信客户端出现。由于播思通信与中国移动的合作关系，已经实现了在 Android 平台上的 Fetion 客户端。对已经实现的 Fetion 客户端进行分析，重用一些软件开发过程中的设计思想，可以对本系统的设计和实现给予很大的帮助。

Fetion 是中国移动通信推出的一款综合即时通信工具。它集成了聊天、交友、互动、娱乐等功能，为用户提供了一个沟通和展示自己的平台。通过 Fetion 可以使用手机和 PC 与对方进行语音聊天、信息交互和发送文件。Fetion 最大的特点就是账号与用户的手机号码绑定，因此，只要是中国移动网络覆盖到的地方，就可以与其它好友通过 Fetion 联系。当然，也只有中国移动用户才能够使用 Fetion，下面就对此客户端程序进行分析。

在 Android 端已经完成的 Fetion 客户端的整体结构图见 3.1：

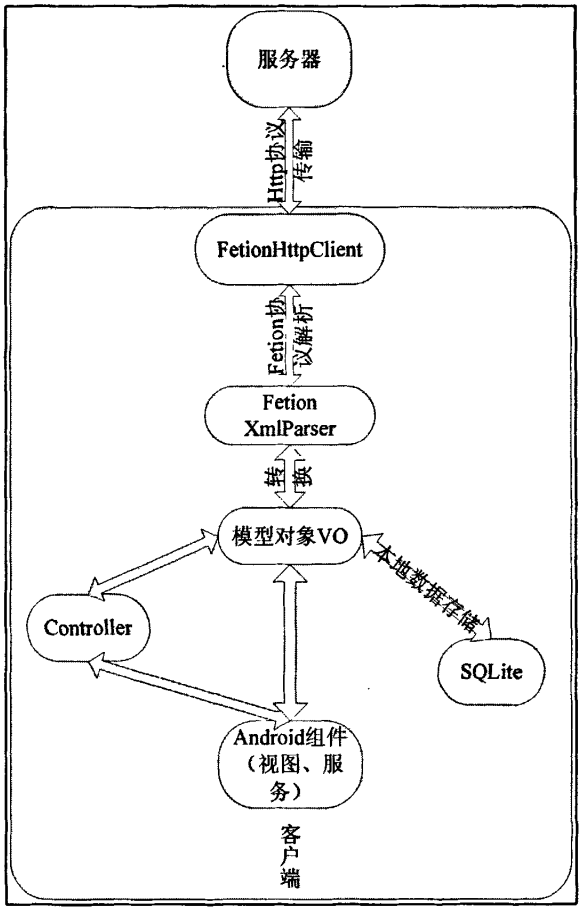


图 3.1 Fetion 客户端体系结构

Figure 3.1 Fetion Client Architecture

如图 3.1 所示，在 Android 平台的 Fetion 客户端中，FetionHttpClient 负责网络通信层的服务器与客户端之间的连接，底层采用 HTTP 协议。Fetion XmlParser 负责 Fetion 协议的解析操作（由于协议的保密性，不对其进行介绍），将从服务器端解析的 Fetion 协议转换为相应的自定义的模型对象，控制器（Controller）、模型对象 VO、Android 组件（视图、服务等）三部分基于 MVC 模式实现，由控制器进行统一的调动，而 SQLite 作为 Android 端嵌入式数据库，负责对模型数据进行缓存，保存用户账户的相关信息。

Andriod 平台的 Fetion 客户端与 PC 上的客户端相比，其实现了 PC 上 Fetion 客户端即时通信的基本功能，摒弃了一些娱乐的功能（交友、语音视频聊天等）。客户端运行截图如图 3.2 所示：


图 3.2 Fetion 客户端运行截图

Figure 3.2 Fetion Client Run ScreenShot

Android 平台上的 Fetion 客户端的主要实现了根据手机号码信息登录 Fetion 客户端，对系统进行相关设置，并与联系人进行信息交互，并实现了最近联系人、陌生人、黑名单以及群组操作模块。

客户端中主要的功能还是与联系人进行信息交互，系统中网络通信使用中国移动通信的 GPRS，基于 Fetion 协议，实现了向好友发送信息，接收好友的信息，查看聊天记录，添加、修改和删除好友等功能，Fetion 中的分组就如图所示，可以将好友拖入陌生人、黑名单（拖入这两个组时好友的行为将受到一定的限制）。另外，系统还提供了群组聊天的功能，用户可以建立聊天群，添加好友至群里，就可以在聊天室里进行信息交互了。其它的功能和市场上类似的 IM 大同小异，这里就不一一介绍了。由于 Fetion 即时通信系统的特殊性，Fetion 要与中国移动通信的手机号码相绑定。在 PC 端可以选择不同的用户登陆，但是在手机端，则只能允许当前手机号码的 Fetion 用户使用。

Fetion 客户端在移动设备，特别是在手机上，是与中国移动通信的手机号码绑定的，这就造成了一定的局限性，致使其它运行商的手机不能登陆该系统。而基于 Jabber 协议则会不同，只要设置相应的 Jabber 服务（服务器地址、端口、服务名称）就可以登陆不同的系统，还可以根据登录服务器扩展的 Transport 与其它协议的系统进行对接，易于在以后进行扩展。

经过了对 Android 平台 Fetion 客户端的详细分析，可以确定在 Android 平台上对基于 Jabber 协议的客户端开发从技术上来说是可以实现的。

## 3.2 系统功能用例分析

本项目根据用户的需求，对即时通信的常用功能进行实现，系统的功能用例图如下所示：

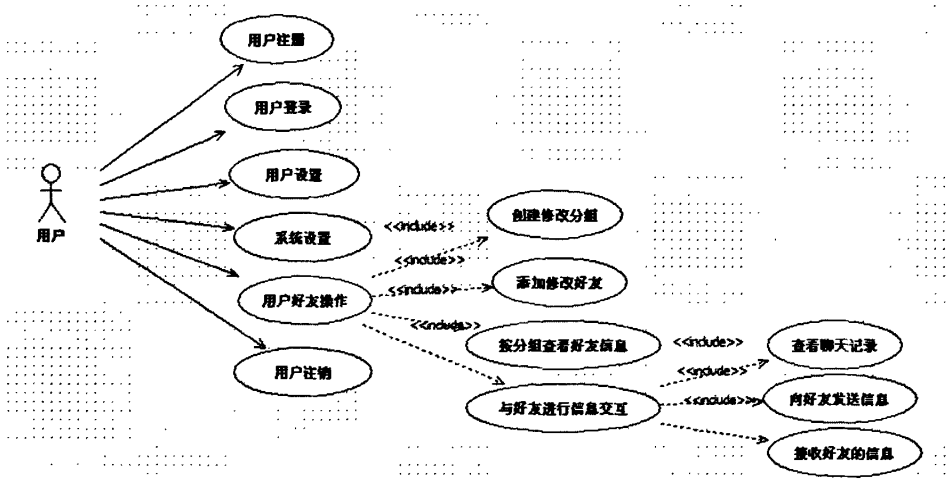


图 3.3 系统用例图

Figure 3.3 System Use-Case Diagram

主要实现功能有：

1. 用户注册：用户如果没有可用的 Jabber ID，可填写相关的个人信息（要注册的 ID、姓名、密码等）进行注册，向服务器提交后服务器返回相应的信息。
2. 用户登录：用户可选择相应的服务（包括服务器地址、服务端口和服务名称）、用户名和密码进行登录即时通信系统。
3. 用户设置：可对用户的基本信息和一些 Jabber 选项进行设置。
4. 系统设置：对本系统的一些选项进行相应的设置（比如宕机后的重连接时间等等）。
5. 查看好友信息：登录成功后，可以按照分组查看所有的好友概要信息（用户名、用户 Jabber ID），并可以查看某个好友的详细信息。
6. 用户创建、修改、删除分组：创建新的分组以对好友进行更好的管理，可以对分组信息进行修改，也可删除现有的分组（但分组内的好友必须为空）。
7. 用户添加、修改、删除好友：根据其它用户的 Jabber ID 添加进自己的好友至某个分组，还可以修改好友的备注信息，并可以删除好友。
8. 用户向好友发送信息：用户登录成功后可给某位好友发送信息，如果好友不在

线，会暂存于服务器中，待其上线后收到信息。

9. 用户接收好友的信息：用户登录成功后会监听好友发送过来的信息，并显示在相应的界面上。
10. 查看聊天记录：用户可查看与某位好友的聊天记录信息。
11. 用户注销：用户注销，退出本系统。

## 3.3 系统非功能性需求分析

- 整体要求

在系统风格方面，应该尽量做到程序结构简单明了，结构条理清晰，功能实用；而在系统界面方面，应该在设计、构思力求精巧，布局要求简单合理，系统整体风格统一。主要细节要求有下面的几个方面：

1. 界面设计友好，提供良好的用户体验，体现企业级应用的特色；
2. 更快的信息访问速度，网络响应速度应该尽可能地满足用户的需求；
3. 更简便、智能化程度更高的程序整体流程；

- 系统扩展

系统要具有良好的可扩展性，便于系统的维护和升级。整个系统采用弹性的架构进行设计，降低需求变更后程序修改所付出的代价。并根据需要提供多种语言支持。

- 错误日志

应用程序应提供易于查看的日志，便于系统的调试、维护和错误定位，系统日志应能保存一定的信息量，并能够记录当前的具体日志信息（时间、程序调用者等）。

- 数据安全

为避免程序中保存的数据被他人盗用，保护用户信息安全，应对程序中持久化的敏感数据进行一定的安全处理。在网络传输过程中，要对传输的数据进行一定的封装处理。在数据库操作过程中，如果事务处理出现故障，进行事务回滚以保证数据的完整性。

## 3.4 软件生命周期过程选型

采用软件工程[14]的管理思想，在项目开发前选用一个合适的软件生命周期模型，是项目成功的关键因素。由于本文中系统需求较为明确，选用用软件工程中的瀑布模型作为开发模型，瀑布模型图如下所示：

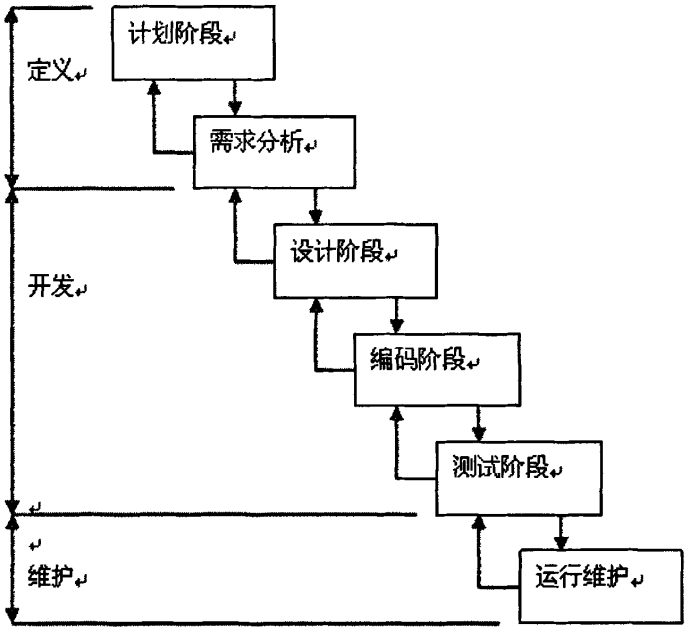


图 3.4 瀑布模型图

Figure 3.4 Waterfall Model Diagram

瀑布模型是当今应用最广泛的一个软件开发模型，其开发过程是通过一系列阶段顺序展开的，从系统需求分析开始直到产品发布，每个阶段都会产生循环反馈，因此，如果有信息未被覆盖或者发现了问题，那么最好返回上一个阶段并进行适当的修改，开发进程从一个阶段流动到下一个阶段。瀑布开发模型的开发阶段主要分为需求分析、系统设计、实现和测试几个阶段。

在本系统开发过程中，在需求阶段通过系统功能用例图进行建模；设计阶段中进行系统总体架构，设计各模块详细设计，确定系统技术路线；系统实现部分中根据系统模块设计进行具体编码实现，在最后对系统功能进行测试工作。

# 4 系统设计

## 4.1 系统总体结构分析

本文的研究目的就是为了实现一种能够为多种移动客户端提供即时通信服务的系统，它可以实现 Android IM 客户端与 PC Jabber 客户端的互相通信，也可以通过各种 Transport 与其它 IM 客户端进行通信（ICQ、MSN、GTalk 等）。系统的体系结构[15]如图所示：

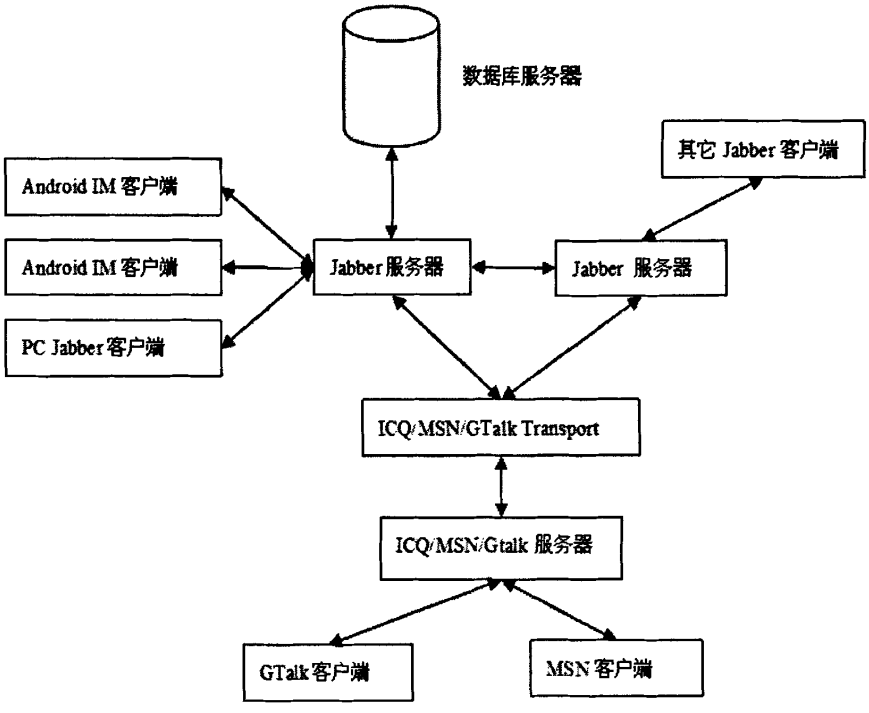


图 4.1 系统体系结构图

Figure 4.1 System Architecture Diagram

Jabber 的服务器端中在整个系统中起着至关重要的作用，Jabber 系统是一个分散的系统，Jabber 服务器分布在网络各处，事实上，由于 Jabber 协议的公开性，网络上也有很多的 Jabber 服务器。其应用服务器端可以由若干个相关组件组成以分别实现系统所要求的一些基本功能，这些基本功能主要包括会话的管理，用户与服务器端的通信，服务器之间的通信，存储用户的个人信息和联系人信息名单，同时也要在服务器端保留用户在离线时所收到的信息，用户注册，用户的身份和权限验证等等，这些也同样涉及到与数据库服务器之间连接并对数据进行操作。

此外，由于即时通信系统的实时性，要求随时要了解对方是否在线，在服务器之间的通信里还要涉及到一个专门的线程模块来负责和其他服务器实时交换用户是否在线的信息。

服务端承担的责任是相当大的，但是客户端也是非常重要的。客户端主要负责通过TCP的Socket套接字通过HTTP协议与服务器端进行通信，发送和接收Jabber协议所用到XML协议信息包，并进行相应的解析以进行相应的操作。由于Jabber协议的特殊性，在这里不会涉及到客户端与客户端之间进行通信。

Jabber协议的最大特点是允许转换的实现，能够兼容各种即时通信系统，和各种即时通信系统互连，可以构建出一个统一的即时通信系统，从而实现与主流的通信软件实现沟通，例如PC中的Pardion、Pidgin，都可以实现多种即时通信工具的互通。支持多协议的即时通信系统[16]如图所示：

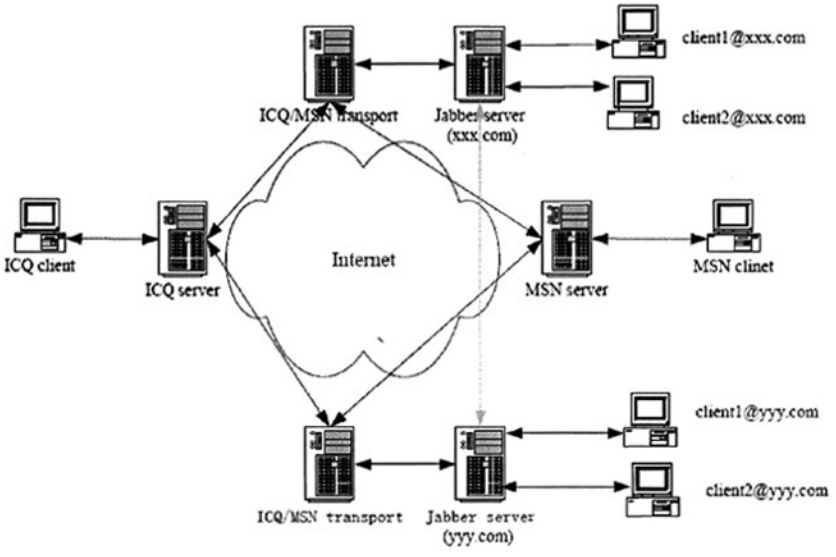


图4.2 支持多协议的即时通信系统图

Figure 4.2 Support Multi-Protocol IM System Structure

负责和其它非Jabber体系的即时通信系统进行通信的组件就是Transport[17]，这也是构成即时通信系统结构的重要组件之一。Transport是在服务器上运行的程序，允许用户登陆其它的即时通信系统，并且接收和发送消息。它负责把来自另一个即时通信系统的消息和包翻译成Jabber格式的XML包。

## 4.2 服务器端设计

Jabber 社区提供了众多的 Jabber 兼容服务器供用户使用，在本系统中，使用了 Openfire(原 WildFire)来搭建服务器。Openfire 是一个跨平台，采用 Java 开发，开源的实时协作（RTC）服务器，基于 XMPP（Jabber）协议。Openfire 的安装和使用相对其它服务器都比较简单，并利用 Web 服务器进行管理，单台服务器可以支持上万并发用户。Openfire 令人难以置信易于安装和管理，而且提供了坚如磐石的安全和性能。

Openfire 配置数据库时，可以使用内嵌的数据库，当然为了提高性能，也可以使用其它的大型数据库进行连接，如 Oracle、Access 等，使用大型数据库时需要提供 JDBC 的支持。在本服务器中使用 MySQL 数据库，数据库中负责存储用户身份数据（Jabber ID 和密码等），好友列表，收到的消息等历史记录持久化的信息。

Openfire 服务器的内核是一个 XMPP 路由器，完成基本组件的之间数据包路由和交换。XMPP 服务器内核负责处理一下公共任务组件：

- 会话管理器：负责客户端会话认证，在线状态，用户联系表等。
- 数据存储器（XDB）：连接数据库系统，保存用户信息、通信日志等。XDB 包括 XDB_file 和 XDB_ldap 子组件，在用户认证时，明文认证发送给 XDB_file，数字认证包括检查 LDAP(Library Directory Access Protocol)用于 XDB——ldap 子组件。
- 连接管理器：管理和客户端之间的连接。
- 服务器连接器：管理 XMPP 服务器之间的连接。
- 传输管理器：建立 XMPP 服务器与非 XMPP 服务器通信。
- DNS 工具：负责域名的解析，查找相应的 XMPP 服务。
- 日志信息管理器：对所有的会话消息进行跟踪，同时服务器端出错时也会记录相应的信息。

Openfire 中添加插件可以为服务器增加新的功能，在本系统中，由于要实现能与其它不同协议的 IM 通信，使用 IM Gateway 插件可以提供与 AOL、ICQ、MSN 等 IM 互相连接的 Transport 支持。

## 4.3 客户端设计

客户端主要实现与服务器端进行通信，向服务器发送和接收相应的请求，完

成即时通信的过程。参照第三章中介绍的Android平台的Fetion协议系统，对整个客户端的架构进行设计，只是在本系统中，使用即时通信协议不同，Jabber 也是基于XML语言进行通信的，只是与Fetion协议相比具体的协议内容、表达方式手段不同。架构设计如图 4.3：

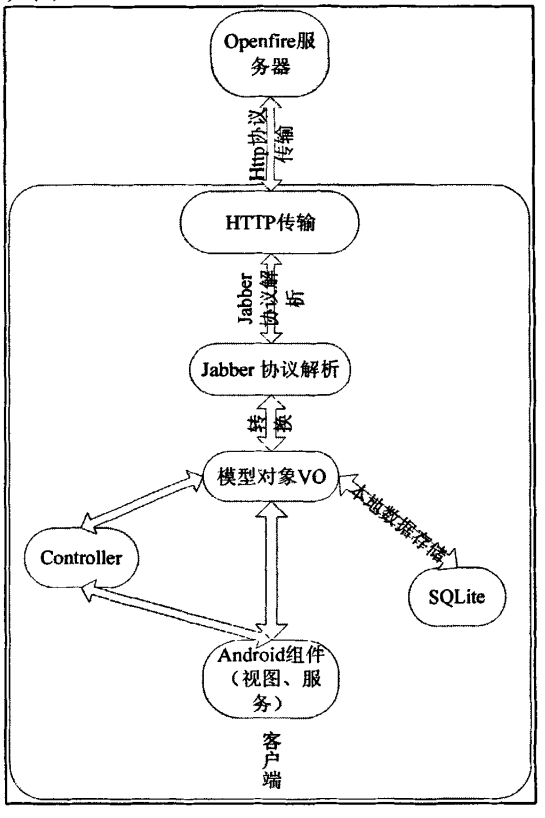


图 4.3 客户端架构设计

Figure 4.3 Client Architecture Design

在本系统中，使用了HTTP协议与服务器进行通信，协议数据中将Jabber数据内容进行封装。HTTP传输模块负责HTTP协议中转，提供数据传输的工作，同时也是会话模型的重要组成部分。Jabber 协议解析模块负责对 Jabber XML 协议进行解析操作，实现 Jabber XML 与系统中用到的模型 VO 对象进行转换。在系统中客户端的内部结构中使用了 MVC 模式[18]，并使用 SQLite 嵌入式数据库保存当前用户的信息。

## 4.4 客户端结构设计

MVC模式是软件设计的典型结构，其广泛应用在交互式系统的开发中，在客

户端中，采用 MVC 模式来进行构建。MVC 模式中应用系统被分成三个部分：模型（Model）、视图（View）和控制器（Controller），职责分离并且非常明确。MVC 模式结构图如下：

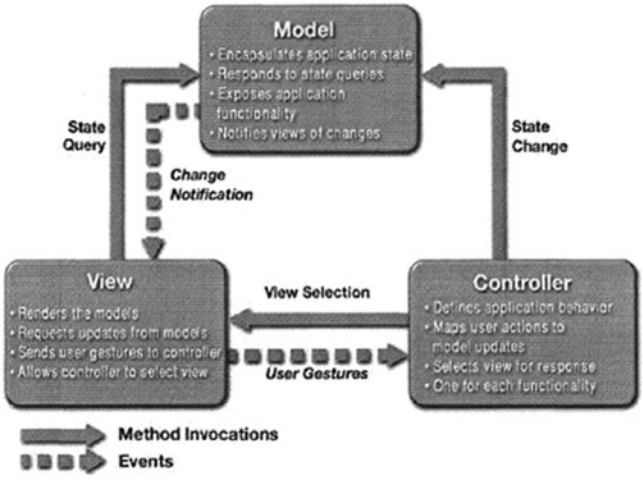


图 4.4 MVC 模式图

Figure 4.4 MVC Diagram

Model 是业务数据和信息处理模块，封装了核心的业务功能和数据。View 向用户展示模型的状态以及控制信息。Controller 作为控制器，负责 View 和 Model 之间的流程控制，一方面，将用户界面（View）的操作映射到具体的模型（Model）上，完成具体的业务逻辑；另一方面，将模型（Model）处理完的业务数据反映到用户界面（View）上。

采用 MVC 模式架构进行设计，可以对整个程序代码进行分层，分为业务逻辑/数据（Model）、界面（View）和控制器（Controller）三层，不仅使程序结构更加清晰，代码更加健壮，而且降低了程序之间的耦合度，提高了模块化程度。这种划分使得模块之间的关系更加清楚，职责更加明确[19]。

## 4.5 数据持久化设计

随着软件应用程序逐渐模块模块化，一种新型数据库会比大型复杂的传统数据库管理系统更适应。嵌入式数据库直接在应用程序进程中运行，提供了零配置运行模式，并且资源占用非常少。

SQLite 是用 C 语言编写的开源嵌入式数据库引擎。它是完全独立的，不具有外部依赖性。SQLite 支持 SQL92 标准，可以在所有主流操作系统上运行，并且

支持大多数计算机语言。SQLite 还非常健壮，可以处理每天负担多达 100,00 次点击率的 Web 站点。SQLite 对 SQL92 标准的支持包括索引、限制、触发和查看。SQLite 不支持外键限制，但支持原子的、一致的、独立和持久 (ACID) 的事务。

由于资源占用少、性能良好和零管理成本，嵌入式数据库将为那些以前无法提供用作持久数据的后端的数据库的应用程序提供了高效的性能。SQLite 之类的嵌入式数据库的易于使用性可以加快应用程序的开发，并使得小型应用程序能够完全支持复杂的 SQL。这一点对于小型设备空间的应用程序来说尤其重要。在本系统中，SQLite 数据库完成对服务端数据进行缓存功能，使得程序可以快速访问本地数据，在必要时才去与服务器同步数据。

本系统中的客户端数据库实体——联系图（E-R 图）如下所示：

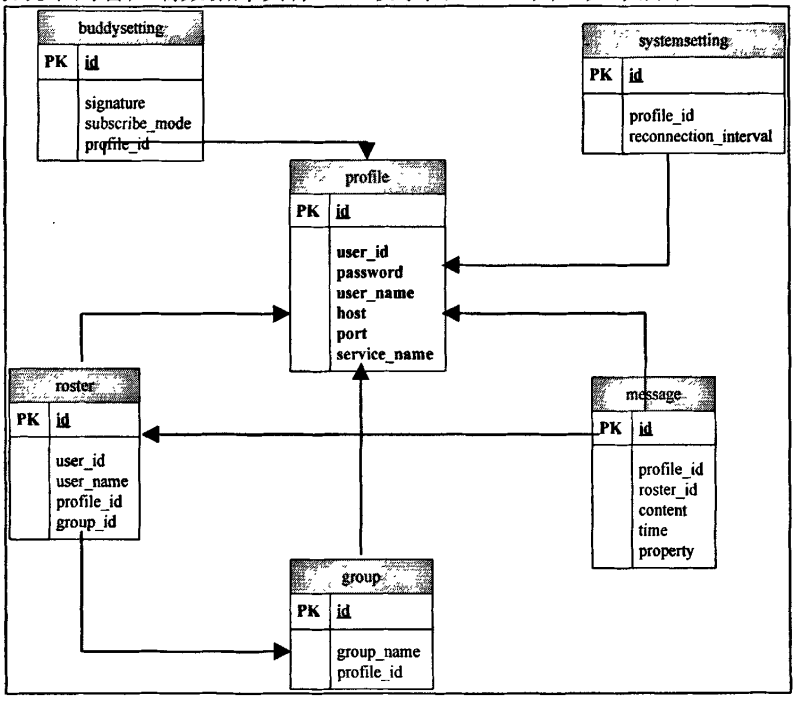


图 4.5 系统数据库 E-R 图

Figure 4.5 System Database E-R Diagram

下面对各个数据库表进行简要介绍：

Profile 表：存储用户配置文件信息；

Roster 表：存储用户的联系人信息；

Group 表：存储用户分组信息；

Message 表：存储用户聊天信息；

BuddySetting 表：存储用户的基本设置信息；

SystemSetting 表：存储系统的设置信息。

## 4.6 会话模型结构设计

Jabber 协议是一个会话协议，在会话过程中，通信的双方都能持续地向对方的发送消息，并且也能够持续地接收对方的消息。Jabber 消息具有异步传输性，发送者和接收者都可以选择在某个时候去处理接收到的消息，故客户端也按照异步消息处理模式进行设计，采用多线程并发，把可能导致阻塞的操作交给其它的线程来处理，这样同时也可以使得无线网络的高延时、低带宽带来的影响最大程度地减小，使系统能够更有效率地运行[20]。

按照即时消息发送的顺序，客户端会话的模型可以分为 HTTP 传输模块、XML 消息解析模块、消息发送和接收模块，其具体关系见图 4.4：

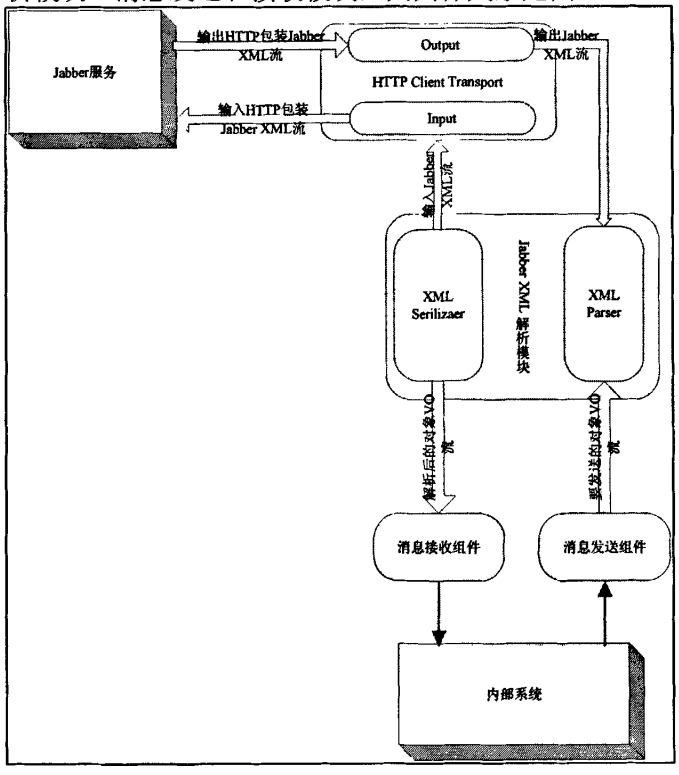


图 4.6 客户端会话模型图

Figure 4.6 Client Conversation Model Diagram

### 4.6.1 HTTP 传输模块设计

HTTP 传输模块负责 Jabber 服务器与 XML 解析模块之间的数据传输以及数据转换。从 XML 解析模块中接收 Jabber 数据，使用 HTTP 协议将其发送给 Jabber 服务，收到响应并将其转换为 Jabber 数据发送给 XML 解析模块。

无线移动应用都是基于移动运营商提供的网络服务基础之上，针对目前我国第三代通信（3G）网络建设正在起步阶段，并且三种制式标准不统一，难以在短时间提供服务。因此，本系统前期应用是基于 GSM 的 GPRS 无线网络，并预留接口以为将来的 3G 网络通信做准备。GPRS 的传输速率平均可达 30kbs，为实现即时消息提供了足够的带宽，并且 GPRS 网络具有覆盖面广、规模大、成熟度高、终端产品丰富、服务优越等特性、可以很好地支撑无线系统的建设工作，另外 GPRS 网络还具有快速连接、永远在线、按流量记费等优点，使得移动即时通信系统使用起来更加方便高效。

目前国际上 GSM 网络通常只有一种 GPRS 接入方式，然而中国移动有 CMWAP 和 CMNET 两种接入方式。CMWAP 和 CMNET 是人为划分的两个 GPRS 接入方式。前者是为 WAP 上网而特别设立的，后者则主要是为 PC、笔记本电脑、PDA 等高端移动设备提供 GPRS 上网服务。虽然它们在实现方式上都是采用 GRRS 作为传输协议，但是由于定位不同，和 CMNET 相比，CMWAP 有部分限制，资费也存在差别。

采用 CMNET 接入方式，移动设备通过网络地址转换设备(NAT)访问外部网络，由于 NAT 是工作在 TCP/IP 网络层次模型的第三层网络层，使得基于这种方式连接的移动设备拥有完全 Internet 的访问能力，对 TCP，UDP 网络传输协议没有任何限制。

采用 CMWAP 接入方式，移动设备通过 WAP 网关访问外部网络，由于 WAP 网关工作在 TCP/IP 网络层次模型的第四层应用层，仅仅实现了 HTTP 代理的功能，并未完成路由、NAT 等局域网网关的功能。这就决定了它在应用上所受到的限制。目前，中国移动的 WAP 网关对外只提供 HTTP 代理协议和 WAP 网关协议的功能。因此采用 CMWAP 接入点时，要求应用程序的网络请求必须基于 HTTP 协议或者 WAP 网关协议。

作为 Jabber 协议默认使用的底层传输协议，TCP 是面向连接的协议，能够提供客户端和服务器之间点到点的连接。如果移动客户端以 CMNET 接入方式通过 TCP 协议连入 Jabber 网络，移动客户端与服务器之间可以建立一条稳定的 TCP 双向通信的会话管道，能直接实现客户端的即时通信。对于多数只支持 CMWAP 接入方式的移动设备来说，我们需要提供 Jabber 协议的 HTTP 代理机制，为移动客户端能够以 HTTP 方式连入 Jabber 网络。本系统中由于移动设备的局限性，故采用 CMWAP 的接入方式。

### 4.6.2 XML 消息解析模块设计

XML（eXtensible Markup Language，可扩充标记语言）是 W3C 组织定义的一种互联网上交换数据的标准。

XML 具有的丰富的表达能力、既对机器友好又对人友好、提倡开放的标准等优点，使得其快速地变成了企业数据交换和集成所选择地的技术，当然，在 Android 平台中程序中的很多地方也都使用了 XML 存储和配置。

相对于文本格式的 XML 片段，对象化的消息更容易被处理。因此，消息解析模块负责用 XML 解析器把文本格式的 XML 片段转换为对象化的消息。

在 Android 平台中，由于受到终端设备性能，网络带宽等条件的限制，就需要采用经过特殊优化的、快速的和轻量级的 XML 解析器。

XML 分析器将基于文本的 XML 文档转换为计算机程序可访问的内存对象。分析 XML 文档有几种方法：

- SAX

SAX 是一种基于事件的分析模型，分析器以线性通过的方式检查完整的文档。SAX 分析过程如图 4.7 所示：

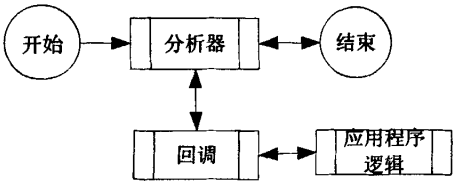


图 4.7 SAX 分析模型图

Figure 4.7 SAX Analyse Model Diagram

SAX 模型的一个最大问题是它是基于推入的：一旦开始分析，分析事件就会被连续推入。这种分析器一次就将完整的 XML 文档分析完毕。开发人员对分析流程无法控制。这样做的效率很低，尤其对移动客户而言。

- XMLPull

XMLPull API 则给予开发人员更多对分析流程的控制。其基于拉出的分析器，可以中断分析过程来处理其他事情，然后再回来继续该分析过程或者中断分析。分析过程如图 4.8 所示：

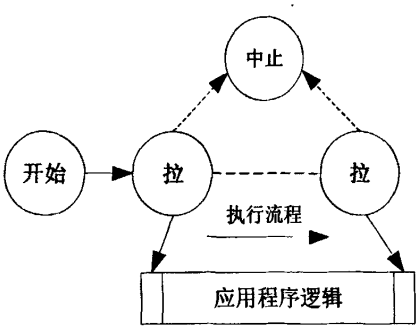


图 4.8 XMLPull 分析模型图

Figure 4.8 XMLPull Analyse Model Diagram

标准的 XML 文档模型 API 是 DOM，但是实践证明，对于移动设备而言，对 DOM 耗费内存较多，支持代价过高，考虑到 XMLPull API 的特点比较适合轻量级开发，文中采用该分析器来对 Jabber 协议进行解析和文档。

### 4.6.3 消息发送和接收设计

- 消息发送

消息发送组件中，首先把消息 VO 转化成为 XML 格式的 Jabber 消息文本，然后通过 HTTP 协议传输，向服务器端提交该文本。发送组件中是客户端主动去向服务器端进行服务请求，在系统中是由业务模型组件完成此部分的工作，视图进行改变后，控制器调度对应的模型组件执行相应的功能操作。

- 消息接收

消息接收组件中，要通过从 HTTP 协议接收到数据，将 XML 格式的 Jabber 协议文本进行解析再进行处理。接收组件一般都是被动地从服务器端得到数据，得到解析后的结果后，根据 MVC 模式，要更新模型（Model）的状态，通知控制器（Controller）对视图（View）进行更新。在这种情况下，使用观察者（Observer）模式可以使得组件之间的依赖性减小，使得低耦合的组件通过调用已知接口中声明的方法进行交互，依赖接口而不是依赖具体类型。

# 5 系统实现

## 5.1 客户端整体结构实现

在系统客户端中，采用 MVC 模式的设计思想，提高了程序的模块化程度[21]。在本系统客户端中，结构图如下：

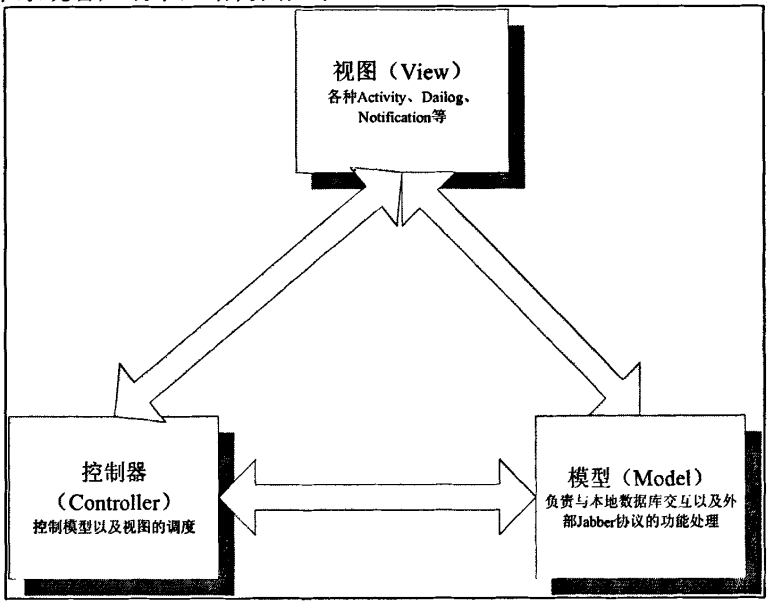


图 5.1 系统客户端 MVC 结构图

Figure 5.1 Client MVC Architecture Diagram

- 视图（View）

视图提供界面功能显示的机制，主要是用户数据的输入界面和对模型数据的显示功能，同时也包括信息的提示功能。

Android 系统中的视图是采用 Activity 组件以及 Dialog 来实现的，而一些 Service 也存在于此。Activity 是最基本的 Android 应用程序组件，在程序中，每个 Activity 都是一个单独的屏幕。每个 Activity 都是一个独立的、从 android.app.Activity 基类继承的类。Activity 会显示由视图组成的用户接口，并对事件（Event）做出响应。大多数的应用是由多屏幕显示组成，当打开一个新的屏幕时，之前一个屏幕会被置为暂停状态并且压入历史堆栈中。用户可以通过回退操回到以前打开过的屏幕。我们可以选择性的移除一些没有必要保留的屏幕，因为 Android 会把每个从桌面打开的程序保留在堆栈中。

而 Dialog 作为对话框，是依赖具体的 Activity 实现的，必须在某个特定的屏

幕中才能显示。一些简短的设置信息，对于某个事件的响应提示信息都设计为Dialog。

- 模型（Model）

模型包括本地数据模型和通信数据模型。其中，本地数据模型用于对本地存储的数据（SQLite 数据库）进行存取、删除、更新等操作；而通信数据模型主要是对与服务器端基于 Jabber 协议的一些功能处理，比如 HTTP 数据交互、Jabber 协议解析等，这些模型部分的实现在下面几节会对其进行详细地介绍。模型也是负责对整个业务流程进行实现的模块，在系统中占据十分重要的位置。

- 控制器（Controller）

控制器主要负责对整个系统的流程进行调度，管理用户界面的逻辑流程，以及用户交互如何影响数据模型和数据模型如何影响用户交互过程，在 Android 开发中，其配置文件 AndroidManifest.xml 扮演控制器的角色，文件中对各种组件视图进行相应的配置，以供模型在改变时对视图的调用。

## 5.2 HTTP 传输模块实现

Android 对网络通信平台的支持还是比较丰富的[22]，除了兼容 J2ME 的 java.net API 外，还提供了一些 Android 平台独有的类 android.net 这个包，但是相比较前两者，其包含的 Apache 实验室的包 org.apache.http 更加强大，对于 HTTP 请求处理很方便，封装得也很好。因此，在本程序中客户端与服务器之间的 HTTP 连接实现使用的就是 org.apache.http 包。

本系统采用的是 GPRS 的 CMWAP 接入方式，中国移动 GPRS 网络目前只有唯一的一个 WAP 网关：10.0.0.172，端口：80，用于 WAP 浏览（HTTP）服务。

使用 Apache 包，先对一些参数进行初始化操作，设置相应的 URL 参数，并创建参数方法（GET、POST 等），并设置固定的 HTTP 头信息，然后执行发送请求，并接收实体数据以及返回的状态代码，释放连接等资源，发送处理代码如下所示：

```
//设置相应的 HTTP 头信息
method.addHeader(HttpConstants.HOST, mHost);
method.addHeader(HttpConstants.USER_AGENT, mUserAgent);
if(DevConFigure.EMULATOR) {
 method.addHeader(HttpConstants.X_UP_CALLING_LINE_ID,RuntimeConFigure.mobile);
```

```
//开始执行相应的操作，发送 HTTP 请求，并获得相应的结果
try {
    HttpResponse response = client.execute(method);
    mStatusCode = response.getStatusLine().getStatusCode();
    HttpEntity entity = response.getEntity();
    mResponseBody = EntityUtils.toByteArray(entity);
    mResponseBodyString =JabberUtil.getUTF8String(mResponseBody);
} catch(Exception e) {
    mStatusCode = FTEvent.E_UNKNOWN;
    mErrorMessage = e.getLocalizedMessage();
    Log.e(DevConFigure.MESSAGING, getClass().getName() + " - " + mErrorMessage);
} finally {
    // 获得返回的状态码，并释放连接
    if(mStatusCode == 0)
        mStatusCode = FTEvent.E_UNKNOWN;
    mSuccessful = mStatusCode == FTEvent.E_OK;
}
```

下面对传输模块中用到的关于 HTTP 部分进行简要说明：

- HTTP 请求

使用 HTTP 进行请求时，分别设置三部分：请求行、消息报头、请求正文。

请求行以一个方法符号开头，以空格分开，后面跟着请求的 URI 和协议的版本，格式如下：Method Request-URI HTTP-Version CRLF。

其中 Method 表示请求方法， 进行请求一般为 GET 或 POST；Request-URI 是一个统一资源标识符；HTTP-Version 表示请求的 HTTP 协议版本；CRLF 表示回车和换行。

- HTTP 响应

在接收和解释请求消息后，服务器返回一个 HTTP 响应消息。HTTP 响应也是由三个部分组成，分别是：状态行、消息报头、响应正文。

状态行格式如下：HTTP-Version Status-Code Reason-Phrase CRLF。其中，HTTP-Version 表示服务器 HTTP 协议的版本；Status-Code 表示服务器发回的响应状态代码；Reason-Phrase 表示状态代码的文本描述。状态代码由三位数字组成，第一个数字定义了响应的类别，有五种可能值：

1xx：指示信息——表示请求已接收，继续处理

2xx：成功——表示请求已被成功接收、理解、接受

3xx：重定向——要完成请求必须进行更进一步的操作

4xx：客户端错误——请求有语法错误或请求无法实现

5xx：服务器端错误——服务器未能实现合法的请求

响应报头允许服务器传递不能放在状态行中的附加响应信息，以及关于服务器的信息和对 Request-URI 所标识的资源进行下步访问的信息。

在本系统中，发送和接收 Jabber XML 数据使用的都是实体报头，请求和响应消息都可以传送一个实体。一个实体由实体报头域和实体正文组成，但并不是说实体报头域和实体正文要在一起发送，可以只发送实体报头域。实体报头定义了关于实体正文和请求所标识的资源的元信息。

上述程序代码中，StringEntity 将请求的 mRequestBodyString 进行封装，并设置实体报头属性，通过 HttpPost 进行发送，接收 HTTP 响应，同样也通过实体报文，根据编码转换为相对应的响应字符串，再通过 XMLParser 进行解析。

## 5.3 XML 消息解析模块实现

Android 平台中就添加了对于 org.xmlpull.v1 API 的支持，负责解析的主要是 XmlPullParser 和 XmlSerializer 两个类，实现对于 XML 的解析以及创建操作。其中 Parser 负责对 XML 文本进行解析[23]，并转换成系统实际需要的 VO 对象；Serializer 负责将请求的 VO 对象生成相应的 XML 文本。

该 API 集的核心是 XmlPullParser 接口。XmlPull 的供应商通过 XmlPullParserFactory 工厂类提供他们自己的 XmlPullParser 实现。下面简单介绍一下控制分析流程的核心方法 next()和 nextToken()。

next()方法将分析器推进到下一个事件。next()方法所能看到的事件安类型是 START_TAG、TEXT、END_TAG 和 END_DOCUMENT；nextToken() 方法给予开发人员更完美的控制能力。它所看到的就是 next()方法所能看到的所有事件。而且还可以报告如下事件：COMMENT、CDSECT、DOCDECL、 ENTITY_REF、PROCESSING_INSTRUCTION 和 IGNORABLE_WHITESPACE。

用 XMLPULL 实现的 XML 解析程序提供了解析各种 Jabber 消息的方法，例如解析片段：

```
public void parse(XmlPullParser parser){
    boolean leave = false;
    do {
     int eventType = parser.next();
     switch (eventType) {
     case XmlPullParser.START_TAG: {
```

```
            String name = parser.getName();
               //判断 XML 标签名，进行匹配，提取相应的信息
            if (JabberConstant.ServerId.equals(name)) {
                //进行相应的信息保存操作……
            } else if (JabberConstant.ParentId.equals(name)) {
                ……
            } else {
                  //没有合适字段，跳过
                ActiveSyncXml.bypassTag(parser);
            }
        case XmlPullParser.END_TAG:
            leave = true;
            break;
        }
        } while (!leave);
    }
```

在 Jabber 协议中提供了 XML 流<stream/>元素以及 XML 流中<message/>、<presence/>、<iq/>3 个顶级元素。在客户端与服务器通信的过程中，会话开始时只发送<stream/>元素的标签开始部分的消息，会话过程中会包含多个<message/>、<presence/>、<iq/>元素格式的消息，只有在会话结束时才会发送</stream>元素的标签结束部分的消息。因此，为了实现上述四种消息的处理，必须在<stream/>元素的标签开始和结束两个位置以及其他元素的结束位置，把解析所得的信息提供给后面的任务链进行处理。在函数中，传递的参数为 XmlPullParser 类型的 parser 对象，操作结束后将 parser 传递给其他对应处理 Jabber 协议的函数。

## 5.4 数据持久化模块实现

Android 的数据（包括 files, database 等）都是属于应用程序自身，其他程序无法直接进行操作。因此，为了使其他程序能够操作数据，可以通过做成 Content Provider 提供数据操作的接口。在本应用中将底层数据封装成了 Content Provider，这样可以有效的屏蔽底层操作的细节，并且使程序保持良好的扩展性和开放性。

Content Provider 就是数据内容的供应者。在 Android 中它是一个数据源，屏蔽了具体底层数据源的细节，在 Content Provider 内部你可以用 Android 支持的任何手段进行数据的存储和操作，比较常用的方式是基于 Android 的 SQLite 数据库。

针对 Android 端的数据操作总结有以下几点：

- **数据库操作**

SQLite 嵌入式数据库比较轻量，没有存储过程之类的繁杂手段，用起来也比较简单。实例化一个 SQLiteDatabase 类对象，通过它的 API 可以执行大部分的操作。Android 中对 DataBase 的使用有一种比较简单的模式，即派生一个 ContentProviderDatabaseHelper 类来进行 SQLiteDatabase 对象实例的获取工作。基本上，ContentProviderDatabaseHelper 类扮演了一个 Singleton 的角色，提供单一的实例化入口点，并屏蔽了数据库创建、打开等细节。在 ContentProvider 中只需要调用 ContentProviderDatabaseHelper 的 openDatabase 方法获取 SQLiteDatabase 的实例就好，而不需要进行数据库状态的判断。

- **URI**

像进行数据库操作需要用 SQL 一样，对 Content Provider 进行增删改查等操作都是通过一种特定模式的 URI 来进行的，URI 的能力与 URL 类似，具体细节可以参考 Android SDK。建立自己的 Content Provider，只需要派生 Content Proivder 类并实现 insert, delete, update 等抽象函数即可。在这些接口中比较特殊的是 getType(uri)。根据传入的 uri，该方法按照 MIME 格式返回一个字符串唯一标识该 uri 的类型。所谓 uri 的类型，就是描述这个 uri 所进行的操作的种类，比如 content://xx/a 与 content://xx/a/1不是一个类型（前者是多值操作，后者是单值），但 content://xx/a/1和 content://xx/a/2 就会是一个类型（只是 id 号不同而已）。

在 ContentProvider 通常都会实例化一个 ContentURIParser 来辅助解析和操作传入的 URI。你需要事先在 static 域内为该 ContentURIParser 建立一棵 URI 的语法树，之后就可以简单调用 ContentURIParser 类的相关方法进行 URI 类型判断（match 方法），获取加载在 URI 中的参数等操作。

- **增删改查**

Content Provider 和所有数据源一样，向外提供增删改查操作接口，这些都是基于 URI 的指令。

进行 insert 操作的时候，你需要传入一个 URI 和 ContentValues。URI 的作用基本就限于指明增减条目的类型（从数据库层面来看就是 table 名），ContentValues 是一个 key/value 表的封装，提供方便的 API 进行插入数据类型和数据值的设置和获取。在数据库层面上来看，这应该是列名与列值的对应。但为了屏蔽 Content Provider 用户涉及到具体数据库的细节，在 Android 的实例中，为每一个表建一个基于 BaseColumn 类的派生类，这个类通常包括一个描述该表的 ContentURI 对象和静态常量值这样的 column 到类数据的对应。当需要进行改变时，可以修改 column 的名字而不需要更改用户上层代码，增加了灵活性。insert 方法如果成功会返回一

个 URI，该 URI 会在原有的 URI 基础上增加有一个 row id，删除和修改操作类似。

删除操作需要传入一个 URI，一个 where 字串，一组 where 的参数（做条件判定），而修改操作会多一个 ContentValues 做更新值。着两个操作 URI 都支持在末尾添加一个 row id。

查询操作的参数是最多的，包括 URI 和一组条件参数。条件参数类型和标准的 sql 类似，包括 sort（排序规则），projection（指定要显示的列名称）之类的。从这些参数到 sql 语句的生成，可以寻求 QueryBuilder 类的帮助，它提供了一组操作接口，简化了参数到 sql 的生成工作。查询返回一个 Cursor，Cursor 是一个支持随机读写的指针，不仅如此，它还提供了方便的删除和修改的 API，是上层对 Content Provider 进行操作的一个重要对象，通过 Cursor 就可以获得要查询的值(类似 JDBC 的 ResultSet)。

- 数据模型

在与界面打交道的 Cursor、ContentResolver 等数据操作层中，采用观察者模式建立数据层与显示层的联系。一个显示层的视图，可以做成某一种观察者注册到 Cursor 或 ContentResolver 等数据中间层中，在实现底层 ContentProvider 中，我们需要特别注意在对数据进行修改操作（包括增删改）后，调用相应类型的 notify 函数，帮助表层对象进行刷新（还有从视图发起的刷新操作）。可以看到 Android 的整体数据显示框架有点像 MVC 的方式。Cursor、ContentResolver 相当于控制层，数据层和显示层的交互通过控制层来掌管，而且控制层很稳定不需要特别定制，通常工作只在定制数据层和显示层空间，还是比较方便和清晰的。

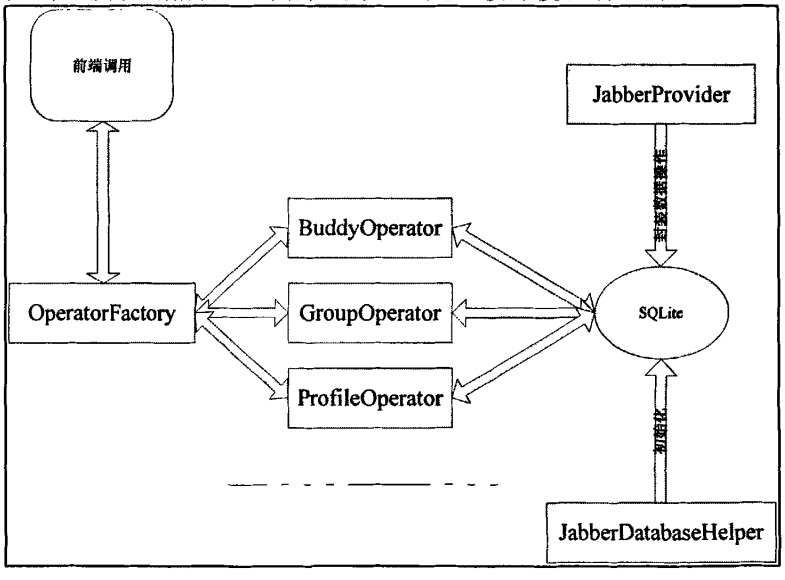


图5.2 持久化模块实现图

Figure 5.2 Persistence Module Realize Diagram

持久化模块实现见上图，在本系统中，使用 Android 中提供的 SQLite 进行数据存储。JabberDatabaseHelper 类继承自 SQLiteOpenHelper，用来对整个数据库文件进行创建和版本控制，建立程序所需要的数据表格。JabberProvider 类继承 ContentProvider，静态块中添加了对应数据表操作的 URI 和相应的 insert，update，delete，query 等实际操作，根据 getType(URI uri)方法返回相应的操作类型，对数据操作进行简单的封装，向外提供基于 URI 的增删改查等基本操作。在上层（数据操作层）应用中，使用 ContentResolver 来进行数据增删改查操作。对于每个数据表均实现相应的数据持久类，对 ContentResolver 的方法进行进一步封装。在外层使用工厂方法模式（Factory-Method）实现前端控制器，这样就只提供一个类供上层应用进行调用，实现了良好的封装性。

## 5.5 前端功能模块实现

前端功能模块就是根据 Android API 进行实现，相当于客户端 MVC 中的视图（有的部分中没有界面部分，如 Service），由控制器控制其流程，与模型部分进行交互。在前面已经介绍过了 Android 端应用程序的组成以及开发，下面就介绍在前端功能模块的结构，如图：

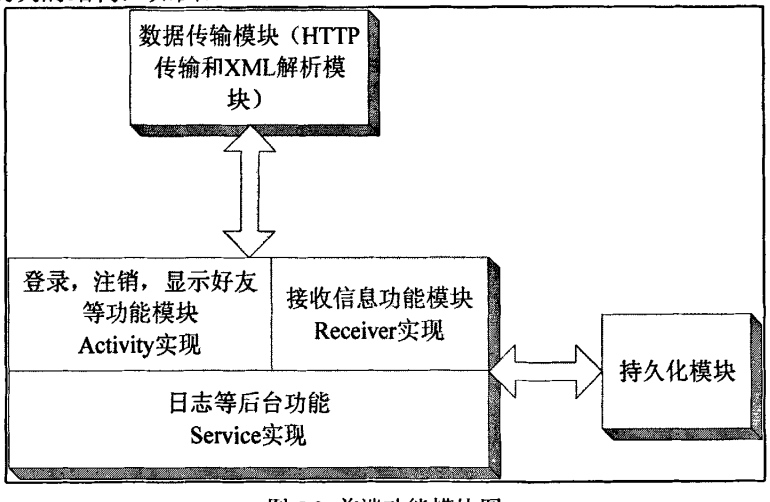


图 5.3 前端功能模块图

Figure 5.3 Front-end Function Module Diagram

Activity 实现登录注销，显示好友，基本设置的功能，在这部分模块中，主动与持久化模块以及数据传输模块进行交互；接收信息模块中被动接收数据传输模块的数据，并进行相应的处理，在这里采用的是观察者模式（Observer）,对感兴趣的上层变化进行监听；日志等操作始终作为 Service 后台实现。

在前端功能模块实现中，使用面向对象的编程思想，将传输模块中的数据对象化，设计出相应的对象进行数据的维护。在登录，注销模块中实现对 Jabber 连接的管理，对好友信息，分组信息抽象出相关类并进行持久化。在接收信息模块中，既要接收用户的即时消息，同时也要处理好友添加等其它信息，对这些信息进行相应的分类管理，并对应各种事件创建相应的 Listener 来监听信息事件，其中包括 MessageListener（即时消息监听），ConnectionListener（连接信息监听），RosterListener（好友花名册信息监听）等。日志等功能由于只是作为后台功能运行，没有界面，故采用 Android 中的 Service 进行实现。

## 5.6 辅助功能实现

### 5.6.1 国际化策略

随着信息的国际化，如何动态地构建一个具有各种不同语言版本的应用程序，成为面向国际应用的企业和个人需要考虑的问题。国际化是使程序在不做任何修改的情况下，就可以在不同的地区和不同语言环境下，按照当地的语言和格式习惯显示字符。

为了使程序能够自适应国际化，实现多国语言，能够根据平台的语言选择相应语言资源对界面进行展示，在本系统中实现了国际化策略。事实上，Google 在设计 Android 框架时就考虑到了该问题，在设计资源相关采用了 MVC 模式，使得代码逻辑和 UI 界面分离更容易维护管理，这里平时在写代码时尽量使用资源文件，不要使用硬编码。目录类似下面：

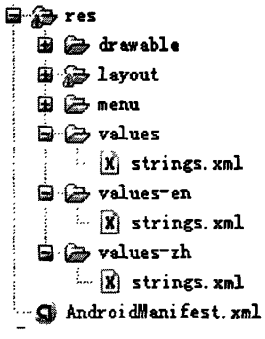


图 5.4 多语言资源目录

Figure 5.4 Multi-Language Resource Catalog

/res/values 为默认文件夹，/res/values-zh 为中文，/res/values-en 为英语，如果要添加其它国家语言以此类推，可以参考 ISO 639-1，其中列出了常用的语言代码。

其中的 string.xml 配置文件中就包含了系统中用到的字符串，系统会根据语言的不同选择不同的配置信息。

### 5.6.2 日志管理

日志信息的管理在软件开发过程中扮演了十分重要的角色。日志管理的好处主要有：首先，记录程序运行时的出错信息，便于软件开发人员分析错误原因，修正 Bug；其次，充当集成开发环境中的调试器的作用，向文件或控制台打印代码的调试信息；最后，监视程序运行的情况，周期性的记录到文件或数据库中，以便日后对其进行统计分析。

在小型 Java 应用程序中，通常采用 System.out.println()语句向控制台打印输出日志信息，而在较大的项目中，往往是使用专门的日志组件类，利用日志类在程序中输出日志信息。在 Android 平台中输出日志，使用 android.util.Log 类，该类提供了若干静态方法，如下：

```
Log.v(String tag, String msg);
Log.d(String tag, String msg);
Log.i(String tag, String msg);
Log.w(String tag, String msg);
Log.e(String tag, String msg);
```

上述方法分别对应 Verbose，Debug，Info，Warning，Error。其中 tag 是一个标识，可以是任意字符串，通常可以使用类名+方法名，主要是用来在查看日志时提供一个筛选条件；msg 是日志的具体信息。程序运行后，并不会在 Eclipse IDE 的控制台内输出任何信息，如果要查看日志，请使用 adb logcat 命令，或在 Eclipse IDE 中切换到 DDMS 视图，查看 Logcat 标签页中的日志信息。对程序添加日志功能后，可以对程序出现的问题进行记录并快速地找到出现异常的原因和代码行，充当程序的调试器。

### 5.6.3 数据安全

在本地的数据保存用的是数据库 SQLite。SQLite 是一个非常小巧的跨平台嵌入式数据库，它的数据以.db 后缀名文件的形式存放在本地磁盘中，但是在其开源的免费版本中却缺少了一个数据库必备的功能，那就是对数据库的加密。SQLite 的数据库文件可以被任何的文本编辑工具打开，如 UltraEdit 或 EditPlus，这一点令很多开发人员和使用者感到很不安。

在这个时代，保护应用程序使用和生成的数据显得异常重要。在设计和实现应用程序时要考虑的一个重要问题就是安全性：防止有价值的数据被剽窃，防止有价值的数据被恶意的攻击者所破坏。本地的 SQLite 数据库文件中的某些信息不能被使用者直接访问出来，例如本系统中使用的 Jabber ID 密码就不能进行明文信息保存至数据库中。

在本系统中，由于要加密的数据不是很多，故可以使用 Java API 中的 javax.crypto 包中的类和接口进行操作。Java 加密扩展即 Java Cryptography Extension，简称 JCE，是 Sun 的加密服务软件，包含了加密和密钥生成等功能，JCE 是 JCA（Java Cryptography Architecture）的一种扩展。JCE 中没有规定具体的加密算法，但提供了一个框架，加密算法的具体实现可以作为服务提供者加入。除了 JCE 框架之外，JCE 软件还包含了 Sun JCE 提供者，其中包括了许多有用的加密算法，比如 DES（Data Encryption Standard）和 Blowfish，在本系统中的加密就使用的是 DES 算法。加密算法用 DESKeySpec 来生成密钥，初始密钥由 8 位任意指定的 byte 数组数据生成。

# 6 系统测试

本系统是 Android 移动平台上应用在无线网络与 Internet 网络之间的 IM 系统，使用户能够方便地与其他移动平台用户或是 PC 端用户进行即时通信的交互，提供了实时查看好友状态信息，与好友进行信息交互收发即时消息等功能。本系统中实现的功能有：

服务器端：使用目前较流行的 Jabber 服务器软件 Openfire，是一个跨平台，采用 Java 开发，开源的实时协作（Real Time Collaboration，RTC）服务器，符合 GPL 版权，基于 Jabber 协议，提供了高度的兼容性，易于安装和管理，并提供了良好的安全和性能。在服务器端配置了 Gateway 插件，以使得与其它 IM（如 MSN、AIM、GTalk）能够进行对接。

客户端：开发的客户端实现了 Basic IM Protocol Suit（JEP——0073）规范要求中提出的必须实现的 Jabber 协议，实现了用户注册和登录，即时消息处理，好友的分组管理，在线状态的监视等等功能。客户端实现了在系统设计中用例图列出的所有功能。

本系统运行在 Android 1.0 SDK 环境中，提供了实现功能必要的用户界面。在客户端中所有界面的流程图如下：

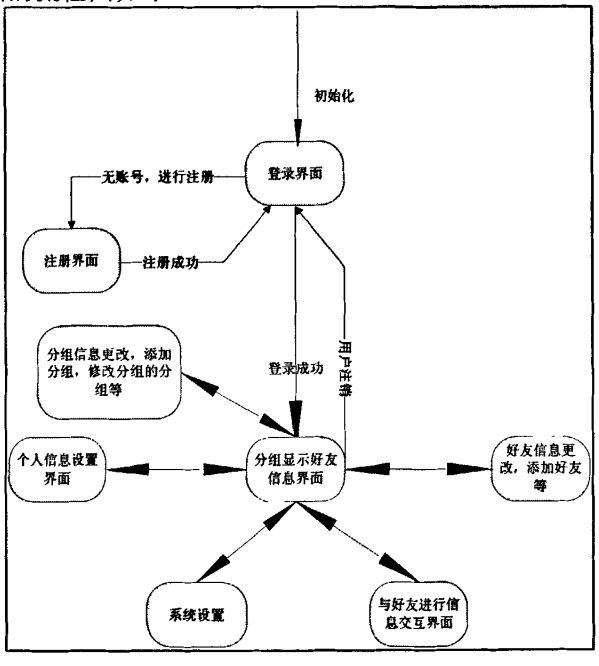


图 6.1 客户端界面流程图

Figure 6.1 Client UI Work Flow Diagram

## 6.1 功能性测试

下面就根据界面的流程图对整个客户端系统的所有界面进行测试（图片中涉及到人名的地方均被屏蔽）。

用户进入系统，将首先进入登录界面，对登录信息（服务器地址，端口，服务名称，用户 Jabber ID，密码，用户显示名称）进行填写。系统将会保存最近一次登录时的信息，在本次登录时可以直接登录或进行修改后再登录。

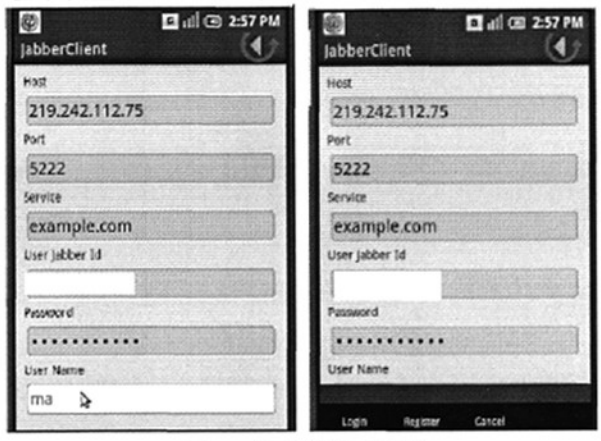


图 6.2 客户端登录界面

Figure 6.2 Client Login UI

用户如果没有 Jabber 账号，则需要在注册后进行登录，注册要填写注册相关信息（申请的 Jabber ID 和密码，以及显示的用户名等），待服务器验证成功后，就可以使用该 Jabber ID 了。注册时，同样要指定要注册的服务器地址以及服务名称，并填写要注册的 Jabber ID，注册成功后，账户方可使用。

用户在登录系统选择菜单 Login 进行登录后，显示所有的好友信息，并对好友进行分组（没有组的好友被分在一个名为 Untitled Group 组中），在好友显示界面中可以进行一系列操作，如分组信息管理（添加分组、修改分组、删除已经为空的分组）、好友信息管理（根据 Jabber ID 添加好友，修改好友信息，添加别名，删除好友）、用户个人信息设置（设置）、系统信息设置、与好友进行信息交互，用户注销等等。

显示所有好友信息的界面如图 6.3，其中中间图显示右键菜单，其中包含了发送信息、查看好友详细信息、修改备注等操作；右边显示单击菜单，包含了系统设置、用户设置等操作。

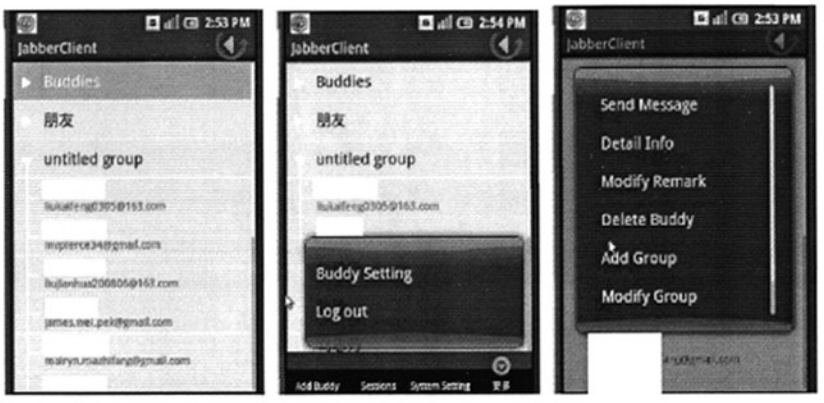


图 6.3 客户端显示好友信息界面

Figure 6.3 Client Show Buddy Info UI

在显示好友界面中，操作是放在右键菜单或功能菜单栏中的。对于分组和组内的好友来说的操作也是不同的（分组右键可以对组进行管理，好友上点击右键可以对好友进行管理，查看好友详细信息界面如图 5.3.4），可以查看用户详细信息以及状态信息（是否可用），如下图所示。其中能够显示用户的 Jabber ID，用户名，分组信息以及状态信息。


图 6.4 查看好友详细信息界面

Figure 6.4 View Friends Detail Info UI

下面在此界面中的功能进行测试。

分组管理：包括添加分组界面、修改分组界面和删除已经为空的分组的功能。添加分组界面中，可以填写分组名称来新建分组，但是如果分组在登录完成后没有添加好友至该分组，则服务器将不会保存此分组信息；修改分组界面中可以对该分组的名称进行修改操作；如果某个分组中的好友列表为空，则可以对该分组进行删除。

好友管理：包括添加好友，修改好友备注信息以及删除某个好友的功能。添加好友界面中，输入好友的 Jabber ID，服务器可以对该好友进行查找，如果存在，

则向该好友发送相应的信息，添加该好友至相应的分组中；修改好友备注信息可以对好友信息进行修改，但只可以更改其备注名称，即在界面中显示的名称；也可以将某个好友删除出分组，此后，该好友的信息不会出现在你的界面中。

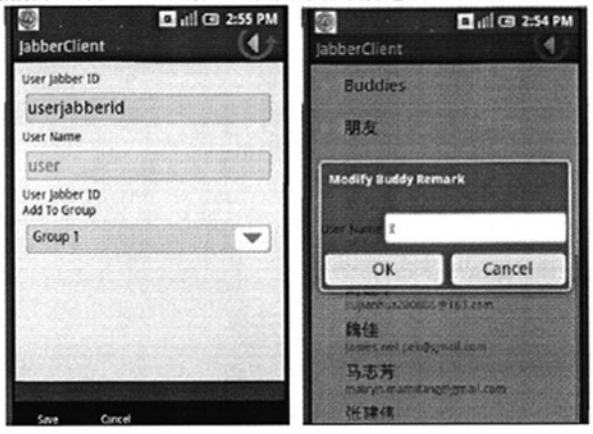


图 6.5 客户端好友操作界面

Figure 6.5 Client Buddy Operate UI

用户个人设置：用户可以对个人信息进行基本设置。本系统中可以设置登录状态（Available、Busy、Don't Disturb 等 Presence 状态信息），设置个人描述信息，以及注册模式（接收所有添加请求、添加提示或拒绝所有的添加请求），具体见图 6.6 的左侧图。

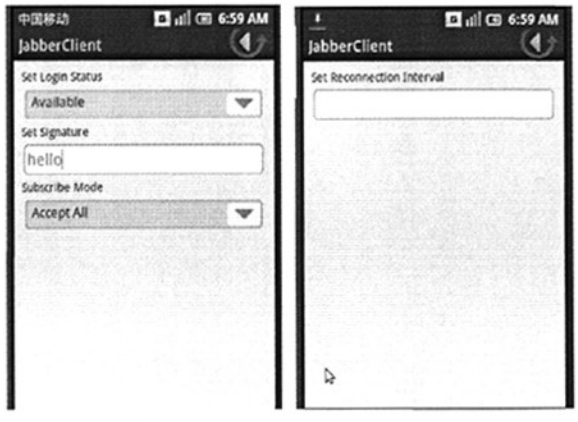


图 6.6 客户端用户设置以及系统设置界面

Figure 6.6 Client User-Setting and Syste-Setting UI

系统设置：用户可以对系统进行必要的设置。本系统中，当前可以设置的只有重新连接的时间，由于移动网络的不稳定性，随时都有可能会连接失败，故设置此选项后可以在指定的时间后进行重新连接。具体界面见图 6.7 的右侧图。

与好友进行信息交互界面，会显示所有与好友的聊天信息，界面中将每组信息分行进行显示，见图 6.7。


图 6.7 客户端与好友信息交互界面

Figure 6.7 Client Inteactive with Buddy UI

用户注销操作：用户进行注销操作，连接关闭，并返回至登录界面。

## 6.2 非功能性测试

对系统要求的其他功能也进行了一些必要的测试工作。系统中界面均基于统一的风格，容易被用户所接受，同时也获得了良好的用户体验，在模拟器上，系统运行速度比较快，并实现了国际化。


图 6.8 实现了国际化的界面

Figure 6.8 Internationalized UI

系统中基于对用户数据安全的考虑，对敏感的数据（比如用户的密码）进行了加密，如图

```
sqlite> select * from profile;
_id|user_id|password|user_name|host|port|service_name
0|clark.mazhiqiang|Sdo3P0d0dkpeds|ma|talk.google.com|5222|gmail.com
sqlite>
```

图 6.9 经过加密的数据显示

Figure 6.9 User Data Encrypted In Sqlite

通过对系统的测试，系统能够在模拟器中运行情况良好，界面风格统一，有着设计良好的程序流程和较好的用户体验，并实现了加密机制和国际化策略，能够满足用户的需求。

# 7 结论

本系统基于固定网络中的即时通信应用和移动网络终端设备的结合，在Android平台上使用Jabber协议实现了即时通信系统。开发过程中以软件工程理论为指导，采用Scrum敏捷开发，用Sprint周期对项目进度进行管理，无论是在项目的控制上还是开发人员协作上都比传统的开发方式具有较大的优势。本项目采用软件工程的开发模式，使用UML进行建模，使分析文档和设计文档都具有良好的可读性和直观性，并对后期扩展进行二次开发具有极大的指导作用。

在系统设计与开发的过程中，主要做了以下工作：

1．对Android平台框架和应用程序的开发进行研究，利用Android框架的优势建立客户端整体模型，并采用持久化方案对服务器数据进行缓存。

2．分析Jabber协议，结合HTTP协议建立系统的会话模型，针对Jabber的异步传输性，采用多线程并发把可能导致阻塞的操作交给其它的线程来处理，可以使得无线网络的高延时、低带宽带来的影响最大程度地减小。

3．根据对研究现状和系统需求的分析，为系统建立合理的整体架构，对系统进行分层设计和实现，使得层次间职责明确，功能分离，并采用松散耦合的方式，易于以后的扩展升级。

即时消息已经成为语音及文本的在线即时通信的主要技术，它的特点决定了其在未来移动商务、在线协作及Internet应用中将扮演更为重要的角色。而随着移动通信技术的迅猛发展，中国3G网络投入商业试运营，为移动即时通信提供了更强有力的支撑平台，实时多媒体技术也会移植到移动即时通信应用中来。集成多媒体的应用有着更强的吸引力，为用户提供更多个性化的服务，将成为未来移动即时通信发展的一个必然趋势。

# 参考文献

[1]臧海峰，张力军.通讯软件发展现状的分析与研究.计算机与数字工程.2008 年第二期.122-124.

[2]Suetterlin P, Thiele O,Knapp H.An OSGi based mobile development overview. ISI. 2008.4.22-26.

[3]Jongbok Byun.Instant Messaging and Presence Technologies for College Campuses. IEEE. 2005.6.4-13.

[4]Peter Saint-Andre.Streaming XML with Jabber/XMPP.IEEE.2006.9.82-89.

[5]张彦，夏清国.Jabber/XMPP 技术的研究与应用.科学技术与工程.2007 年第七卷第六期.51-53.

[6]于少山，卡米力，毛依丁.基于 XML 的即时通信系统的研究与实现.重庆邮电大学学报.2007 年 6 月.59-61.

[7]刘丹.08-09 年中国企业 IM 市场发展综述.赛迪顾问.

[8]潘振香，翟明岳.Jabber 协议在即时通信系统中的应用.网络安全技术与应用.2007.10.80-81.

[9]Google.Android SDK-windows-1.0_r1 Documentation..

[10]姚昱曼，刘卫国.Android 的架构与应用开发研究.计算机系统应用.2008 年第 11 期.110-112.

[11]Jabber 官方文档.http://www.jabber.org.

[12]MOHD HILMI HASAN, ZURAIDAH SULAIMAN, NAZLEENI SAMIHA HARON, AZLAN.Enabling Interoperability Between Mobile IM and Different IM Applications Using Jabber.ACM.2007.8.51-55.

[13]潘永高，钟亦平，张世永.基于网关的 J2ME Jabber 系统研究.计算机工程.2005 年第 31 卷第 19 期.108-110.

[14]张海藩.软件工程导论（第四版）.清华大学出版社.2003.11.

[15]Jerry Gao, Ph.D., Mansi Modak, Satyavathi Dornadula, and Simon Shim, Ph.D. Mobile Jabber IM: A Wireless-Based Text Chatting System.IEEE.2004.10.43-48.

[16]Marco Ughetti，Tiziana Trucco，Danilo Gotta.Development of agent-based, peer-to-peer mobile applications on Android with JADE.ACM.2008.7.287-294.

[17]樊燕红，谭香.基于 XMPP 的即时通信网关应用研究.计算机技术与应用.2007 年第 10 期.123-124.

[18]张弛，陈建明，刘敏，赵秋霞.基于 J2ME 平台的 IPv6 Jabber 系统实现.计算机应用研究.2006 年第 5 期.169-171.

[19]Jerry Gao, Ph.D., Mansi Modak, Satyavathi Dornadula, and Simon Shim, Ph.D. Mobile Jabber IM: A Wireless-Based Text Chatting System.IEEE.2004.10.43-48.

[20]姚昱曼，刘卫国.Android 与 J2ME 平台间即时通信的研究与实现.计算机系统应用.2008 年第 12 期.118-120.

[21]Yen-Wen Lin,Li-Su Kuo,Jhih-Siang Wang,Wei-Ting Shiu.P2P-Based Instant Messaging for Wireless Community Networks.IEEE.2005.1.15-19.

[22]孙卫琴.Java 网络编程精解.电子工业出版社.2007.3.

[23]张容，苗放，李刚.XMPP 及其在即时通信系统的文字通信模块中的应用.重庆工学院学报.2008 年 2 月.92-95.

# 作者简历

姓名：马志强 性别：男 出生日期：1986年1月17日
籍贯：河北省 最后学历：硕士 毕业院校：北京交通大学

教育经历：

2007.9——2009.7 北京交通大学 软件学院
专业及学位： 软件工程 工程硕士
2003.9——2007.7 北京交通大学 计算机与信息技术学院
专业及学位： 计算机科学与技术 工学学士

实习经历：

2008.9——2009.3 播思通讯技术有限公司
在实习期间，参与了 Android 平台上 Fetion、ActiveSync 相关产品以及本项目的开发工作。

# 基于Android平台即时通信系统的设计与实现

作者：马志强
学位授予单位：北京交通大学

本文链接：http://d.g.wanfangdata.com.cn/Thesis_Y1578068.aspx
授权使用：湖南大学(hunandx)，授权号：8c4458f6-cfc8-4db5-9382-9ea600af120a
下载时间：2011年3月14日