**华东交通大学**

**毕业设计（论文）**

**题目： 基于Android的即时通讯软件的设计与实现**

|  |  |  |  |
| --- | --- | --- | --- |
| **学 院:** | **软件学院** | | |
| **专 业:** | **软件工程+桥梁工程** | **班 级:** | **2015-2** |
| **学生姓名:** | **马海峰** | **学 号:** | **2015211003000230** |
| **指导教师:** | **丁琼** | **完成日期:** | **2019年5月10日** |

|  |
| --- |
| **毕业设计（论文）诚信声明**  **本人郑重声明：所呈交的毕业设计（论文）是我个人在导师指导下进行的研究工作及取得的研究成果。就我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表和撰写的研究成果，也不包含为获得华东交通大学或其他教育机构的学位或证书所使用过的材料。**  **如在文中涉及抄袭或剽窃行为，本人愿承担由此而造成的一切后果及责任。**  本人签名  导师签名  **2019 年 5 月 10日** |

**华东交通大学毕业设计（论文）任务书**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 姓名 | 马海峰 | | | 学号 | | 2015211003000230 | | | 毕业届别 | 19届 | | 专业 | 软件工程+桥梁工程 |
| 毕业设计（论文）题目 | | | | | 基于Android的即时通讯软件的设计与实现 | | | | | | | | |
| 指导教师 | | 丁琼 | | | | 学历 | | 硕士研究生 | | | 职称 | | 讲师 |
| **具体要求：**  1、设计目的与内容  当今社会，社交聊天软件必不可少，微信就是一个成功的社交聊天软件，此项目就是仿照微信的模式，基于环信的即时通讯云SDK，开发一个新的社交聊天软件，其主要功能包括，注册登录登出，添加联系人，与联系人进行聊天，申请群聊，与多个联系人进行群聊。  2、设计要求与成果   1. 通过文献阅读和英文翻译，掌握研究课题相关的现状和关键技术。（符合毕业要求2.2、10.3） 2. 掌握系统工作原理和软件工程相关规范和技术要求。（符合毕业要求12.2、11.2） 3. 完成系统需求分析、概要设计、详细设计及编码测试。（符合毕业要求3.2、5.4）   撰写毕业设计论文、答辩，表达自己的观点。（符合毕业要求10.1、10.2）  3、应收集的主要参考文献  [1] Google.AndroidTraining（中文版）.Google，2012.1  [2] 严蔚敏.数据结构（C语言版）（第二版）.人民邮电出版社，2015.2  [3] 王珊，萨师煊.数据库系统概论（第五版）.高等教育出版社，2014.9  **进度安排：**   |  |  |  |  | | --- | --- | --- | --- | |  | 设计各阶段内容 | 时间 | % | | 1 | 选题和资料收集阶段 | 第7学期18～20周 | 5 | | 2 | 分析计划阶段 | 第8学期1～2周 | 10 | | 3 | 设计阶段 | 第8学期3～5周 | 25 | | 4 | 实现和测试阶段 | 第8学期6～9周 | 40 | | 5 | 毕业论文写作、查重、答辩 | 第8学期10～14周 | 20 | | 6 | 英文资料翻译 | 自定 |  |   指导教师签字：  2019年1月10日 | | | | | | | | | | | | | |
| 题目发出日期 | | | 2019.1.11 | | | | 设计（论文）起止时间 | | | 2019.1.14-2019.6.9 | | | |
| 学院意见：  同意发布题目  毕业设计领导小组组长签章 | | | | | | | | | | | | | |

**华东交通大学毕业设计（论文）开题报告书**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 课题名称 | 基于Android的即时通讯软件的设计与实现 | | | | |
| 课题来源 |  | 课题类型 | AX | 导师 | 丁琼 |
| 学生姓名 | 马海峰 | 学号 | 2015211003000230 | 专业 | 软件+桥梁工程 |
| **一、开题报告内容**  1、目的和意义  当今社会，社交聊天软件必不可少，微信就是一个成功的社交聊天软件，目前来看，聊天软件单一且拥有广大的前景，现在世面上最受欢迎的就是微信已经QQ，此项目就是仿照微信的模式，基于环信的即时通讯云SDK，开发一个新的社交聊天软件，其主要功能包括，注册登录登出，添加联系人，与联系人进行聊天，申请群聊，与多个联系人进行群聊。此项目力求给用户一个纯粹的聊天工具，没有五花八门，没有花里胡哨，唯一的功能就是聊天，可以单聊，可以群聊，没有广告，干净快速便捷。  2、研究现状和发展趋势  腾讯QQ、微信等社交软件在不断扩大其新用户的过程中，也需要完善软件周边服务，维护老用户。各软件在发展过程中积极与第三方平台合作，例如，与淘宝等购物平台开展的免注册登录购物体验，给用户带来了方便，但同时也存在安全隐患，仍然有很大一部分大学生对此表示担心。只有消除这些用户的担忧，才能给软件的发展创造更大的空间。  　　各社交软件在发展过程中不断拓展配套应用服务，满足顾客的不同需求，提升软件综合服务能力，进而提升软件的市场竞争力。主流社交软件的各项服务都有相应规模的受众群体，并提高相应满意度，这是软件发展必不可少的一部分。  　　三、影响各软件发展的因素  　　社交网络的本质是人际关系网络，而社交软件则是为社交网路提供载体。社交软件的存在是为了满足人们的需求，包括社交需求、娱乐需求及快捷需求等。那么社交软件在发展过程中一方面需要不断深入地了解用户需求，另一方面则需要不断完善自身服务以满足用户需求。主流社交软件分别具有各自的定位和市场针对性，腾讯QQ是基于强社交关系，用户在使用过程中注重交流；新浪微博则是基于弱社交关系，主要是服务于热点新闻的传播，覆盖面大，传播路径短。那么相应地，各社交软件基于自身定位，加强与第三方应用的合作，不断完善配套应用的服务质量，抓住用户需求，将在市场竞争中获得优势。  　　四、主流社交软件的优劣势分析  　　（一）优势  　　第一，价格优势。目前国内社交软件的一大特点就是免费。软件的大多数功能免费，用户在使用过程中只需要一定的流量费即可实现方便快捷的交流。  　　第二，功能优势。各主流社交软件都有自己独特的定位，满足用户的不同需求。无论在细节还是功能方面都具有自己的特色，正是这些特色吸引了软件早期用户，这是基础。随着软件的不断升级优化，每个版本都在提升品质，创新服务，在功能上有着确定的优势。  　　第三，规模优势。主流社交软件在发展过程中积累了庞大的用户群体，在时间上占据了早期发展的优势。其已有的用户群体能够支撑起软件的运营，这就减少了软件发展的阻力，能够推动软件突破推广用户的局限，集中资源致力于软件的开发和服务。  　　第四，资源优势。不同的主流社交软件都具有各自的资源优势，腾讯具有庞大的用户资源，新浪具有强大的媒体资源。这些无法复制的资源奠定了社交软件的发展地位，新兴软件在竞争过程中会受到很大的竞争阻力。  　　（二）劣势  　　第一，管理难度。随着主流社交软件的不断发展，其存在的用户群不断扩展，相应地会滋生各种管理问题，许多用户问题无法得到及时解决，需要投入更大的管理成本。  　　第二，模糊定位。社交软件在发展过程中会不断发展其服务功能，功能的不断丰富会导致定位的模糊，很容易与其他社交软件产生重叠，从而导致推广不善。例如，腾讯微博与新浪微博的重叠。  　　第三，竞争压力。社交市场不断发展，用户需求被不断细分。新兴功能的社交软件层出不穷，很多独具特色的社交软件异军突起。不断发展的新兴社交软件正在抢占社交市场，主流社交软件因其规模上的限制，很难在短期内进行转型，从而存在很大的竞争风险。  3、设计方案  工具：Android Studio，Genymotion，Genymotion是一款专业的手机模拟器，可以设置多个不同型号的手机，用于不同版本的软件部署  语言：JAVA，Android，手机移动开发需要有JAVA基础，语法与结构与JAVA相似。   1. 进度安排  |  |  |  |  | | --- | --- | --- | --- | |  | 设计各阶段内容 | 时间 | % | | 1 | 选题和资料收集阶段 | 第7学期18～20周 | 5 | | 2 | 分析计划阶段 | 第8学期1～2周 | 10 | | 3 | 设计阶段 | 第8学期3～5周 | 25 | | 4 | 实现和测试阶段 | 第8学期6～9周 | 40 | | 5 | 毕业论文写作、查重、答辩 | 第8学期10～15周 | 20 | | 6 | 英文资料翻译 | 自定 |  | | | | | | |

课题类型：（1）A—工程设计；B—技术开发；C—软件工程；D—理论研究；

（2）X—真实课题；Y—模拟课题；Z—虚拟课题

（1）、（2）均要填，如AY、BX等。此部分可以附页

**华东交通大学毕业设计（论文）开题报告书（续）**

|  |
| --- |
| 5、主要参考文献  [1] Google.AndroidTraining（中文版）.Google，2012.1  [2] 严蔚敏.数据结构（C语言版）（第二版）.人民邮电出版社，2015.2  [3] 王珊，萨师煊.数据库系统概论（第五版）.高等教育出版社，2014.9   1. **方法及预期目的**   1、研究方法（手段）  通过Android Studio实现开发，开发过程中部署Genymotion模拟器来查看开发过程中不同阶段所实现的功能，语言基础是JAVA语言。最终实现一个可运行，可操作的手机APP，实现注册登录登出，添加联系人，与联系人进行聊天，申请群聊，与多个联系人进行群聊等功能。  2、预期目的  设计实现一个可运行，可操作的手机APP，实现注册登录登出，添加联系人，与联系人进行聊天，申请群聊，与多个联系人进行群聊等功能。  **三、指导老师意见**  同意开题  指导教师签名： 日期：2019.3.15 |

**华东交通大学毕业设计(论文)评阅书（1）**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **姓名** |  | **学号** |  | **专业班级** |  |
| **毕业设计(论文)题目** | |  | | | |
| **指导教师评分表：**   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | 课程目标 | 毕业要求指标点 | 权重系数 | 评判标准及分值 | | 得分 | | 目标1 | 2.2:通过文献研究分析复杂软件工程问题，以获得有效结论。 | 0.4 | 文献检索及分析总结能力，论点正确，研究方案合理。 | 20 |  | | 目标2 | 3.2:能够通过科学分析，根据基本思路和方案，设计满足特定需求的系统、组件或软件过程，并根据需求完成数据结构和算法的设计，并在设计中体现创新意识。 | 0.25 | 系统分析、设计、开发方法合理，能实现预期功能，有一定难度，工作量满足要求。 | 20 |  | | 5.4:能够采用恰当的方法和工具对软件进行测试和验证，能够应用文档等形式呈现软件系统解决方案和成果。 | 0.4 | | 11.2:熟悉软件生命周期的各个环节，并能够在软件工程问题分析中，合理运用工程管理与经济决策方法。 | 0.3 | | 目标3 | 10.1:针对软件工程领域复杂工程问题，能与业界同行及社会公众进行口头、书面的沟通和交流，准确陈述和表达自己的观点。 | 0.4 | 论文内容正确，结构严谨，格式规范，文献翻译数量质量符合要求。 | 20 |  | | 10.2:针对专业问题，能够使用外语进行一定的口头和书面等方式的表达和交流。 | 0.6 | | 10.3:能通过阅读和交流，了解软件工程及计算机领域的国际发展趋势、研究热点。 | 0.4 | | 目标4 | 12.2:具备终身学习的知识基础，掌握自主学习的方法，了解拓展知识和能力的途径，能够自主学习，适应社会和职业发展需要。 | 0.3 | 对前人工作有改进或有独特见解，有一定应用价值。 | 20 |  | | 工作态度与平时表现。 | 20 |  | | 合计： | | | | 100 |  |   **指导教师评语：**  **指导教师签字：**  **2019年 5 月 日** | | | | | |

**华东交通大学毕业设计(论文)评阅书（2）**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **姓名** |  | **学号** |  | **专业班级** |  |
| **毕业设计(论文)题目** | |  | | | |
| **评阅人评分表：**   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | 课程目标 | 毕业要求指标点 | 权重系数 | 评判标准及分值 | | 得分 | | 目标1 | 2.2:通过文献研究分析复杂软件工程问题，以获得有效结论。 | 0.4 | 文献检索及分析总结能力，论点正确，研究方案合理。 | 20 |  | | 目标2 | 3.2:能够通过科学分析，根据基本思路和方案，设计满足特定需求的系统、组件或软件过程，并根据需求完成数据结构和算法的设计，并在设计中体现创新意识。 | 0.25 | 系统分析、设计、开发方法合理，能实现预期功能，有一定难度，工作量满足要求。 | 30 |  | | 5.4:能够采用恰当的方法和工具对软件进行测试和验证，能够应用文档等形式呈现软件系统解决方案和成果。 | 0.4 | | 11.2:熟悉软件生命周期的各个环节，并能够在软件工程问题分析中，合理运用工程管理与经济决策方法。 | 0.3 | | 目标3 | 10.1:针对软件工程领域复杂工程问题，能与业界同行及社会公众进行口头、书面的沟通和交流，准确陈述和表达自己的观点。 | 0.4 | 论文内容正确，结构严谨，格式规范，文献翻译数量质量符合要求。 | 30 |  | | 10.2:针对专业问题，能够使用外语进行一定的口头和书面等方式的表达和交流。 | 0.6 | | 10.3:能通过阅读和交流，了解软件工程及计算机领域的国际发展趋势、研究热点。 | 0.4 | | 目标4 | 12.2:具备终身学习的知识基础，掌握自主学习的方法，了解拓展知识和能力的途径，能够自主学习，适应社会和职业发展需要。 | 0.3 | 对前人工作有改进或有独特见解，有一定应用价值。 | 20 |  | | 合计： | | | | 100 |  |   **评阅人签字：**  **2019年 5 月 日** | | | | | |

**华东交通大学毕业设计(论文)评阅书（3）**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **姓名** |  | **学号** |  | **专业班级** |  |
| **毕业设计(论文)题目** | |  | | | |
| **答辩小组评分表：**   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | 课程目标 | 毕业要求指标点 | 权重系数 | 评判标准及分值 | | 得分 | | 目标1 | 2.2:通过文献研究分析复杂软件工程问题，以获得有效结论。 | 0.4 | 论点正确，研究方案合理。 | 20 |  | | 目标2 | 3.2:能够通过科学分析，根据基本思路和方案，设计满足特定需求的系统、组件或软件过程，并根据需求完成数据结构和算法的设计，并在设计中体现创新意识。 | 0.25 | 设计合理，难易程度、工作量满足要求，能实现预期功能。 | 20 |  | | 5.4:能够采用恰当的方法和工具对软件进行测试和验证，能够应用文档等形式呈现软件系统解决方案和成果。 | 0.4 | | 11.2:熟悉软件生命周期的各个环节，并能够在软件工程问题分析中，合理运用工程管理与经济决策方法。 | 0.3 | | 目标3 | 10.1:针对软件工程领域复杂工程问题，能与业界同行及社会公众进行口头、书面的沟通和交流，准确陈述和表达自己的观点。 | 0.4 | 答辩准备充分，时间把握合理，回答问题有理论依据，思路清晰，表达准确。 | 20 |  | | 10.2:针对专业问题，能够使用外语进行一定的口头和书面等方式的表达和交流。 | 0.6 | | 论文内容正确，结构严谨，格式规范，文献翻译数量质量符合要求。 | 20 |  | | 10.3:能通过阅读和交流，了解软件工程及计算机领域的国际发展趋势、研究热点。 | 0.4 | | 目标4 | 12.2:具备终身学习的知识基础，掌握自主学习的方法，了解拓展知识和能力的途径，能够自主学习，适应社会和职业发展需要。 | 0.3 | 对前人工作有改进或有独特见解，有一定应用价值。 | 20 |  | | 合计： | | | | 100 |  |   答辩小组组长（签字）：  **2019年 5 月 日** | | | | | |

**华东交通大学毕业设计（论文）答辩记录**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **姓名** |  | **学号** |  | **专业班级** |  |
| **毕业设计(论文)题目** | |  | | | |
| **答辩组成员（签字）**： | | | | | |
| 答辩记录：  记录人（签字）：  2019年 5 月 日 | | | | | |

**华东交通大学毕业设计(论文)成绩**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **姓名** |  | **学号** |  | **专业班级** |  |
| **毕业设计(论文)题目** | |  | | | |
| **答辩委员会意见：**  **同意以上评定，根据前面三项得分，按25:15:60的比例评定总成绩为**   |  |  | | --- | --- | | **等级** |  |   **进入二次答辩（毕业设计成绩由二次答辩的最终成绩决定）**    □ **评优组**  □ **合格组**    **答辩委员会主任签字：**  **2019年 5 月 日（学院公章）** | | | | | |

注：“等级”用优秀、良好、中等、及格、不及格五级制。优秀：90~100分；良好：80~89分；中等：70~79分；及格：60~69分；不及格：60分以下。

基于Android的即时通讯设计和实现

摘 要

随着科技的发展和不断进步，人们已经从飞鸽传书，到驿站邮件，再到有线电话，到手机，到现在的即时通讯，交流方式发生了翻天覆地的变化，越来越方便，越来越快捷，例如微信，已经成为我们生活中不可或缺的一部分，几乎每个人的交流都是通过微信来完成的，这也从另一方面表现出即时通讯带来的变化。

现如今的即时通讯，大多基于手机，手机正是人们随身携带的，必备的物品，而即时通讯，强调的就是即时和通讯。即时，意思就是马上，发出的消息不会延迟，，不会等待。而通讯，就是沟通，交流，

此系统使用Android作为开发平台，Android studio作为开发工具，MySQL数据库作为用户信息的存储，基于环信的SDK，涉及多线程，使用网络传输和文件流进行文件传输。这个聊天软件包括客户端和服务器。服务器使用环信服务器作为所有客户端信息传输的中转站，可以实时显示软件注册用户和在线用户的数量。用户之间的聊天将被反馈到服务器端，每个用户将离线显示在服务器端。客户端包括新用户的注册、注册用户的登录以及找回密码和修改昵称头像和密码，聊天包括私人聊天和群聊。

本文首先通过对网络聊天软件的背景、研究意义和可行性的分析，阐述了网络聊天软件的发展前景；其次，通过需求分析，阐述了系统的总体结构和功能；通过系统设计，介绍了此即时通讯软件的总体结构和功能模块。对系统进行了规划设计，根据系统设计建立了数据库，建立了表格。通过对系统实现的介绍，详细描述了系统具体功能模块的实现过程。本系统严格遵循软件开发的基本过程，从分析设计到系统实现和测试。

关键词：即时通讯；网络传输；Android

**Design and Realization of Human Resource Management System Based on .Net**

**Abstract**

It is great significant to promote the management of the enterprise with not only the changing of environment but also the increasing of competition. However, the management of human resource always plays a crucial part in the enterprise management and the excellent one will lead to a powerful enterprise. Furthermore, as the computer, especially Internet technology have been developed promptly, an open, efficient, and widely expanded human resources management system which has become one of the indispensable manners to enhance the core competitiveness of enterprises should be built urgently.

In this paper, first of all, the background of the human resources management and its system has been analyzed in order to clear out the intention of this research. Secondly, in accordance with the requirements of software engineering project the demands of the enterprise business and its system which are based on the analysis of process and problem in business have been put forward and then the system has been modeled in this chapter. Thirdly, the system including software applications architecture, network architecture and business process is founded on OOA and presented by applying UML and three-layer architecture with B/S in Chapter III. In the fourth chapter, the database involving the modeling (ER diagram) and the logical devise has been schemed. The development employing ADO.NET and AJAX as well as the test of system which is interactive, practical and scalable have been conducted in the fifth chapter.

This study has made great efforts on the architecture design and the interaction to achieve a strongly expanded, highly code-reused, conveniently maintained. This system which is practicable and expedient is so different from the traditional B/S structure based one which is less humanized and capable. In this paper, the main developed parts focus on the management of basic data, personnel alteration, payroll and attendance and systems itself.

Keywords: Human resources management system; Three-layer architecture; AJAX

目 录

[1 绪 论 1](#_Toc318664910)

[1.1 研究的背景及意义 1](#_Toc318664911)

[1.1.1 选题的背景 1](#_Toc318664912)

[1.1.2 国内外研究现状 1](#_Toc318664913)

[1.1.3 研究的意义 2](#_Toc318664914)

[1.2 系统目标 2](#_Toc318664915)

[2 需求分析 3](#_Toc318664916)

[2.1 功能需求 3](#_Toc318664917)

[2.1.1 子系统/模块说明 3](#_Toc318664918)

[2.1.2 功能需求描述 3](#_Toc318664919)

[2.2 非功能需求 3](#_Toc318664920)

[2.2.1 外部接口需求 3](#_Toc318664921)

[2.2.2 性能需求 3](#_Toc318664922)

[2.2.3 其它需求 3](#_Toc318664923)

[3 总体设计 5](#_Toc318664924)

[3.1 运行环境 5](#_Toc318664925)

[3.2 基本处理流程 5](#_Toc318664926)

[3.3 模块结构 5](#_Toc318664927)

[3.4 外部接口 5](#_Toc318664928)

[3.5 内部接口 5](#_Toc318664929)

[4 数据库设计 6](#_Toc318664930)

[4.1 概念结构设计 6](#_Toc318664931)

[4.1.1 设计思路 6](#_Toc318664932)

[4.1.2 E-R图 6](#_Toc318664933)

[4.2 逻辑结构设计 6](#_Toc318664934)

[4.2.1 设计思路 6](#_Toc318664935)

[4.2.2 逻辑模型 6](#_Toc318664936)

[4.3 物理结构设计 6](#_Toc318664937)

[4.3.1 存取方式 6](#_Toc318664938)

[4.3.2 存储结构 6](#_Toc318664939)

[5 界面设计 7](#_Toc318664940)

[5.1 界面关系图或工作流图 7](#_Toc318664941)

[5.2 界面设计成果 7](#_Toc318664942)

[5.2.1 主界面 7](#_Toc318664943)

[5.2.2 子界面 7](#_Toc318664944)

[6 详细设计 8](#_Toc318664945)

[6.1 系统主要功能模块介绍 8](#_Toc318664946)

[6.2 模块一设计 8](#_Toc318664947)

[6.2.1 模块一算法描述 8](#_Toc318664948)

[6. 2.2 模块一程序流程图 8](#_Toc318664949)

[6. 2.3 模块一关键类说明 8](#_Toc318664950)

[6.3 模块二设计 8](#_Toc318664951)

[6.3.1 模块二算法描述 8](#_Toc318664952)

[6.3.2 模块二程序流程图 8](#_Toc318664953)

[6.3.3 模块二关键类说明 8](#_Toc318664954)

[…… 8](#_Toc318664955)

[7 编码 9](#_Toc318664956)

[7.1 代码实现与核心算法 9](#_Toc318664957)

[7.2 代码优化分析 9](#_Toc318664958)

[8 测试 10](#_Toc318664959)

[8.1 测试方案设计 10](#_Toc318664960)

[8.1.1 测试策略 10](#_Toc318664961)

[8.1.2 测试进度安排 10](#_Toc318664962)

[8.1.3 测试资源 10](#_Toc318664963)

[8.1.4 关键测试点 10](#_Toc318664964)

[8.2 测试用例构建 10](#_Toc318664965)

[8.2.1 测试用例编写约定 10](#_Toc318664966)

[8.2.2 测试用例设计 10](#_Toc318664967)

[8.2.3 关键测试用例 10](#_Toc318664968)

[8.2.4 测试用例维护 10](#_Toc318664969)

[9 总结与展望 11](#_Toc318664970)

[9.1 设计工作总结 11](#_Toc318664971)

[9.2 未来工作展望 11](#_Toc318664972)

[谢 辞 12](#_Toc318664973)

[参考文献 13](#_Toc318664974)

[附录A 外文翻译—原文部分 15](#_Toc318664975)

[附录B 外文翻译—译文部分 16](#_Toc318664976)

[附录C 软件使用说明书 17](#_Toc318664977)

[附录D 主要源代码 18](#_Toc318664978)

[附录E 相关图表 19](#_Toc318664979)

1 绪 论

## 1.1 研究的背景及意义

1.1.1 选题的背景

即时通讯是一个终端连往一个即时通讯网路的服务。即时通讯不同于e-mail在于它的交谈是即时的。大部分的即时通讯服务提供了presence awareness的特性──显示联络人名单，联络人是否在线上与能否与联络人交谈。

随着计算机技术特别是互联网技术的不断发展，信息技术对人类社会生活的影响越来越大，正日益改变着我们的生活方式。人，作为社会的主体，显得尤为重要，而沟通，作为人的基本能力，更是重中之重，及时的通讯，不但能提高人们的生活体验，更是能够提高工作效率。因此，现如今的通讯更讲究及时和便捷，不管是公众日常生活，还是公司内部任务部署，都离不开即时通讯。可以说即时通讯已经渗入到我们生活的方方面面，我们已经离不开即时通讯。

1.1.2 国内外研究现状

当今，国际上对网络通信系统研究的较好的公司有思科，Sun，Ms等公司，思科主要研究的是底层的传输，Ms和Sun公司研究的应用层。而国内，在应用层上的网络应用软件目前发展异常的火爆，如Tencent的即时通讯，从某种程度上来说，代表了目前国内的最高水平。

现在国内的即时通讯工具按照使用对象分为两类：一类是个人IM，如qq，微信，淘宝的旺旺等待。QQ的前身OICQ在1999年2月第一次推出，激几乎垄断了中国在线即时通讯软件市场。现如今，同样是腾讯公司的微信，已经成为即时通讯行业的巨头。

现在的即时通讯软件有几个大的特点：多功能性，是一个综合信息平台，即时通讯的功能从一个单纯的交流沟通工具，演进为综合信息平台，集交流、资讯、娱乐、搜索、游戏、购物等多功能于一体。广告商、媒体、游戏厂商等越来越多的成员加入到这一产业，共同推动即时通讯的综合化平台发展。商务化，即时通讯从最初的个人生活聊天应用逐渐发展成为办公交流、客户服务沟通、电子商务等重要的使用工具。产品多元化，软件产品及服务层出不穷。可以说，现如今的即时通讯软件技术已经成熟，而且开发一个即时通讯软件也不再是一个多难的事情，开发即时通讯的辅助包网上多如牛毛，但是开发一个成熟的好用的即时通讯软件并没有那么容易，就像微信，那是一个团队不断创新，不断实践，不断优化出来的产品。除了微信，如钉钉，旺旺等通讯软件也同时占据国内即时通讯的一席之地。

### 1.1.3 研究的意义

即时通讯软件未来发展会更加强调互联网应用的整合，各IM软件之间的互联互通已成大势所趋，即时通信在聚集了一定的人气之后，将逐渐成为一种时尚潮流，进而成为一种社会文化现象，现在，即时通讯企业都想尽办法将语音和视频业务拓展到传统的固定电话和移动电话上，从最基本的语音服务，到P2P的数据传输、电话会议、视频会议等。未来即时通讯业务承载的设备趋于多样化，用户通过PC、手机、平板电脑以及其他设备等都可以使用即时通讯，手机定制将成为即时通讯厂商推广产品的重要手段之一。

## 1.2 系统目标

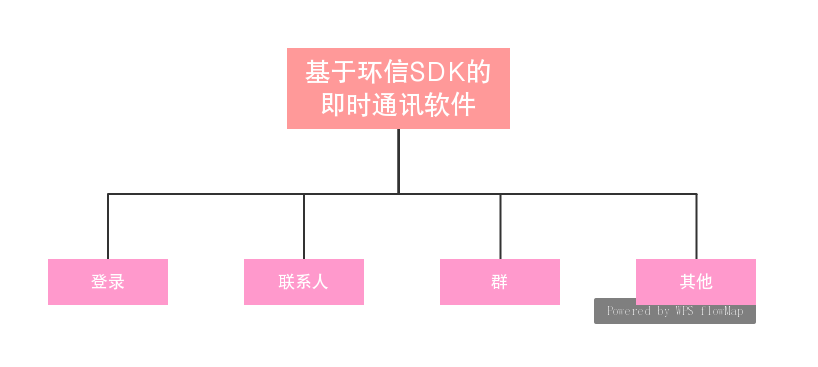
设计实现一个可运行，可操作的手机APP，实现注册登录登出，找回密码，添加联系人，与联系人进行聊天，申请群聊，邀请好友进群，与多个联系人进行群聊修改昵称、头像密码等功能。

2 需求分析

## 2.1 功能需求

2.1.1 子系统/模块说明

本程序实现的在Android系统下的，即时通讯软件，所实现的模块包括：欢迎页面，登录模块，会话模块，联系人模块，群模块，添加模块以及设置模块。所包括的具体功能有：注册，登录，登出，添加及删除好友，建立与解散以及推出群聊，单人与单人聊天，多人建群聊天，其中聊天包括发送文字，发送语音，发送图片。

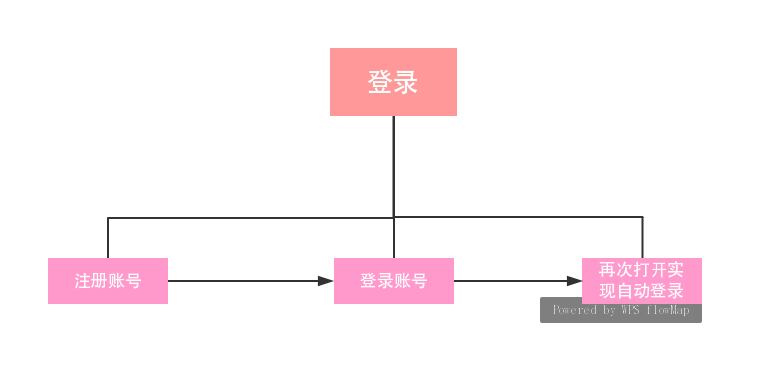


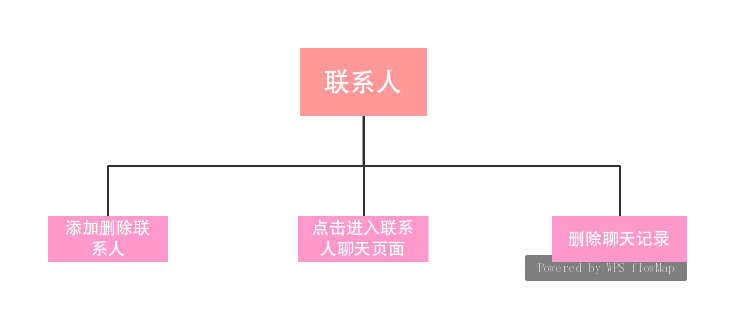
### 2.1.2 功能需求描述

具体模块的功能描述：

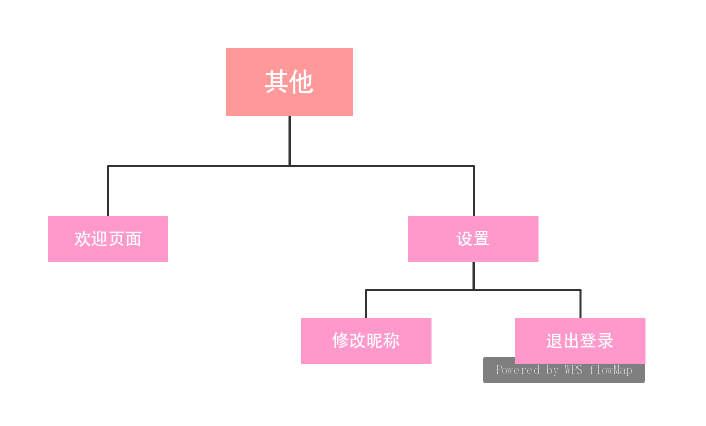
1. 欢迎页面没有实现功能，设置一张图片，缓冲两秒进入登录模块
2. 登录模块：包括注册功能，找回密码功能与登录功能
3. 会话模块：列表展示所有会话，包括单聊与群聊，点击进入聊天页面
4. 联系人模块：包括列表展示所有联系人，实现长按删除联系人，点击进入与联系人聊天页面
5. 群模块：包括建立群，列表展示所有群，点击进入群聊页面，以及邀请列表用户加群和删除群成员，退出群解散群等功能
6. 添加模块：实现搜索用户，发送好友邀请功能
7. 设置模块：显示用户昵称与头像，设置用户昵称，设置用户头像，以及更改密保问题和退出登录功能

### 2.1.3 功能详述结构图

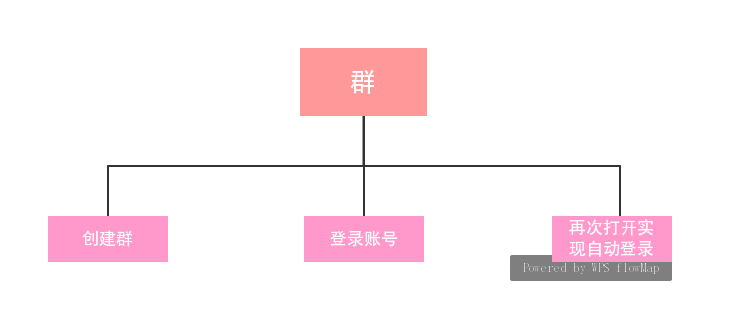
登录操作的功能结构

联系人的功能结构

其他模块的功能操作



群模块的功能操作



## 2.2 非功能需求

### 2.2.1 外部接口需求

读写SD卡权限，网络权限，调用摄像头权限

### 2.2.2 性能需求

1）会话数据即时传输与更新

2）数据库数据的同步更新

3）页面信息随数据库信息即时更新

4）即时提醒用户与APP交互操作

5）需请求网络权限

3 总体设计

## 3.1 运行环境

3.1.1：客户端：Android手机

3.1.2：网络协议：TCP/IP

3.1.3：服务器端：环信服务器

3.1.4：网络协议：TCP/IP

## 3.2 基本处理流程

进入APP后，在登录页面有三个操作选择，（一）直接输入账号密码进行登录操作，（二）注册后登录，（三）如果忘记密码，可找回密码

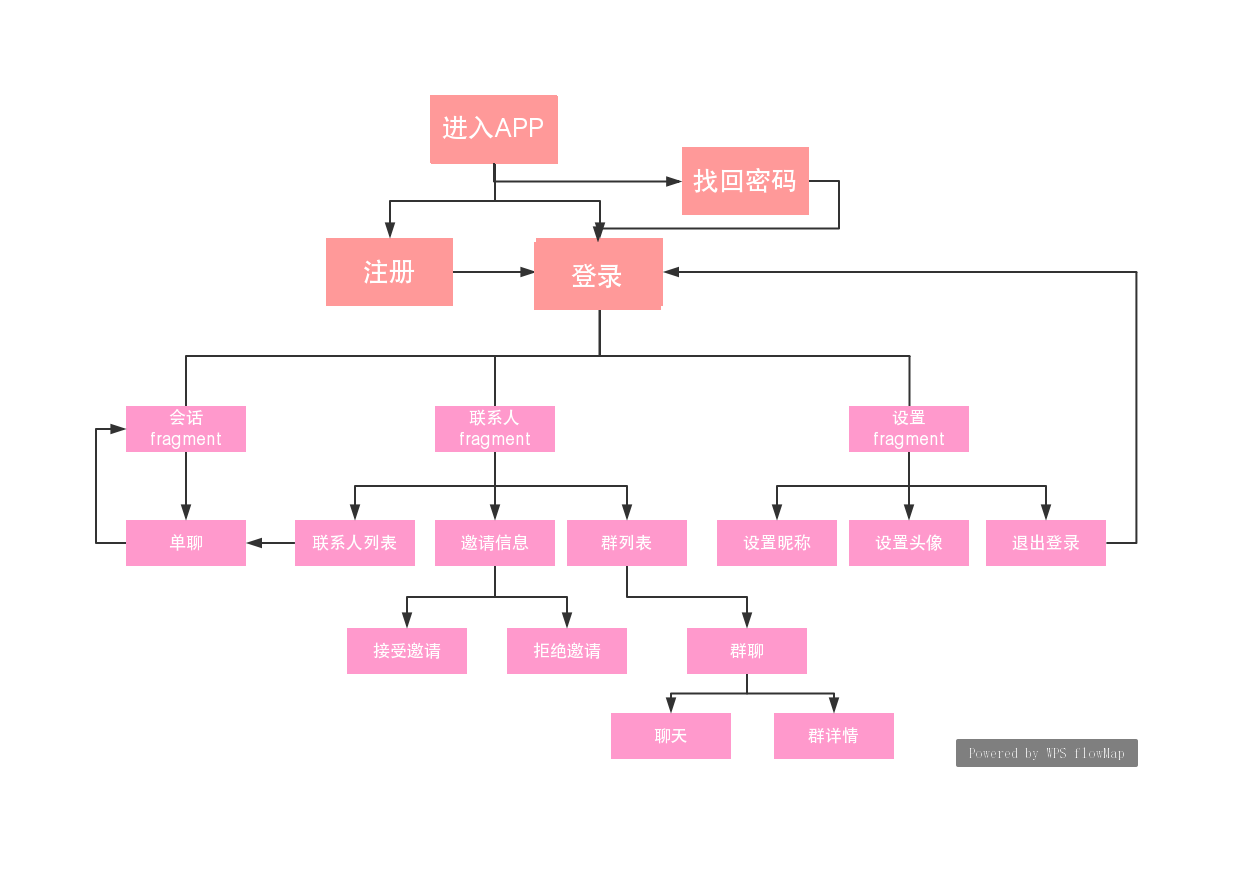


图3.2-1 总体流程图

成功登录以后，可进行后续一系列操作。

## 3.3 模块结构

3.3.1Activity信息：

1. 添加好友Activity：

所包含的操作：初始化各控件，搜索用户，添加好友

1. 聊天Activity：

所包含的操作：初始化各控件，监听点击事件

1. 群详情Activity：

所包含的操作：初始化GridView，加载GridView适配器，监听退群/解散群的点击，发送退群/解散群的广播

1. 群列表Activity：

所包含的操作：初始化ListView，加载ListView适配器，获取并加载群信息

1. 邀请信息列表Activity：

所包含的操作：初始化ListView，加载ListView适配器，监听接受和拒绝点击事件

1. 登录Activity：

所包含的操作：初始化各控件，用户登录以及存数据到数据库

1. 主页面Activity：

所包含的操作：初始化各控件，切换fragment

1. 新建群Activity：

所包含的操作：初始化各控件，创建群

1. 找回密码Activity：

所包含的操作：初始化各控件，找回密码

1. 修改昵称Activity：

所包含的操作：初始化各控件，修改昵称，监听修改的点击事件

1. 注册Activity：

所包含的操作：初始化各控件，注册账号，监听确定和取消的点击事件

1. 群成员邀请选择页面Activity：

所包含的操作：初始化各控件，从本地数据库获取所有好友，监听保存的点击事件

1. 修改头像Activity：

所包含的操作：初始化各控件，修改头像，监听确定的点击事件

1. 欢迎界面Activity：

所包含的操作：设置图片，显示两秒进入登录页面

3.3.2数据库：

使用MySQL和SQLiteDatabase存储数据。

存储模块：

1. 用户信息表：

表名，用户姓名，用户ID，用户昵称，用户头像

所包含的操作：添加用户信息进数据库，从数据库获取用户信息，根据用户ID获取用户头像，存储用户头像进数据库，更新用户昵称

1. 联系人信表：

表名，用户姓名，用户ID，用户昵称，用户头像，用户是否好友

所包含的操作：获取多个联系人，获取单个联系人，保存联系人，删除联系人

1. 邀请信息表：

表名，用户姓名，用户ID，群名称，群ID，邀请原因，邀请状态

所包含的操作：新增邀请信息，获取邀请信息，转换邀请状态，移除邀请信息，更新邀请信息

1. 注册人信息表：

表名，用户ID，用户昵称，用户头像，用户密码

所包含的操作：新增注册信息，判断是否是已注册的用户，获取注册人信息，获取注册人的头像

3.3.3各模块流程图

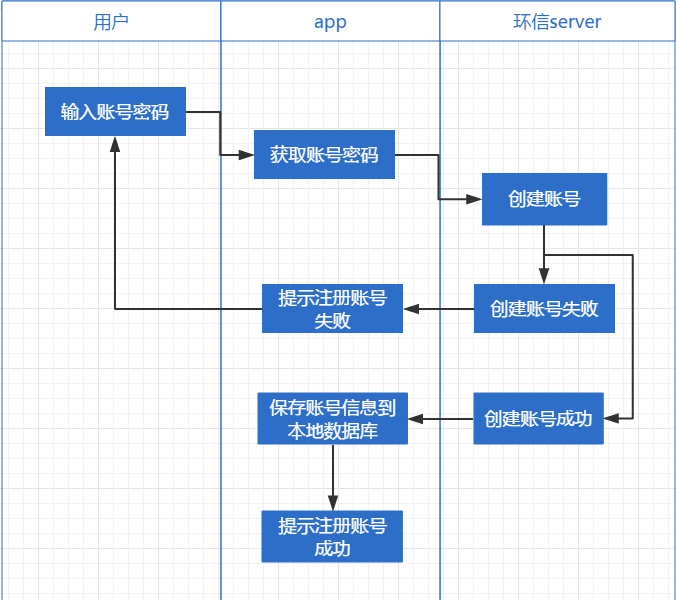
注册账号：

图3.3-1 注册流程图

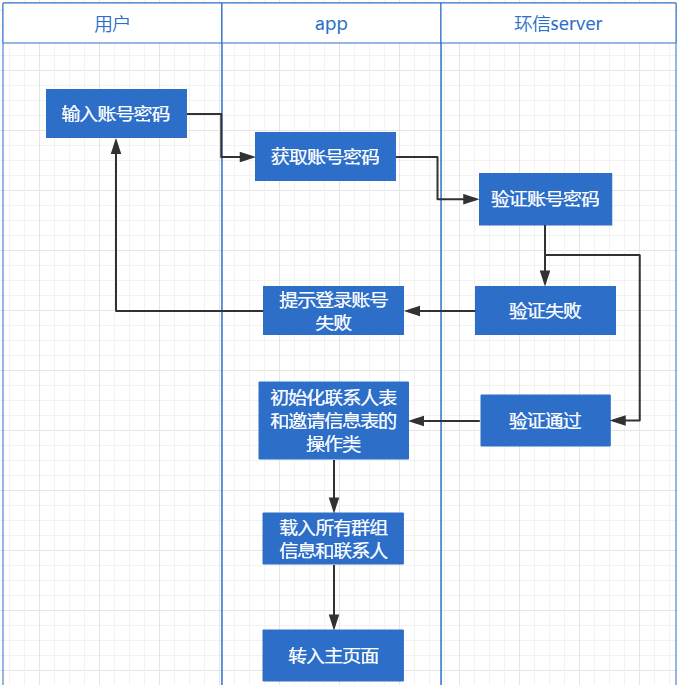


图3.3-2 登录流程图

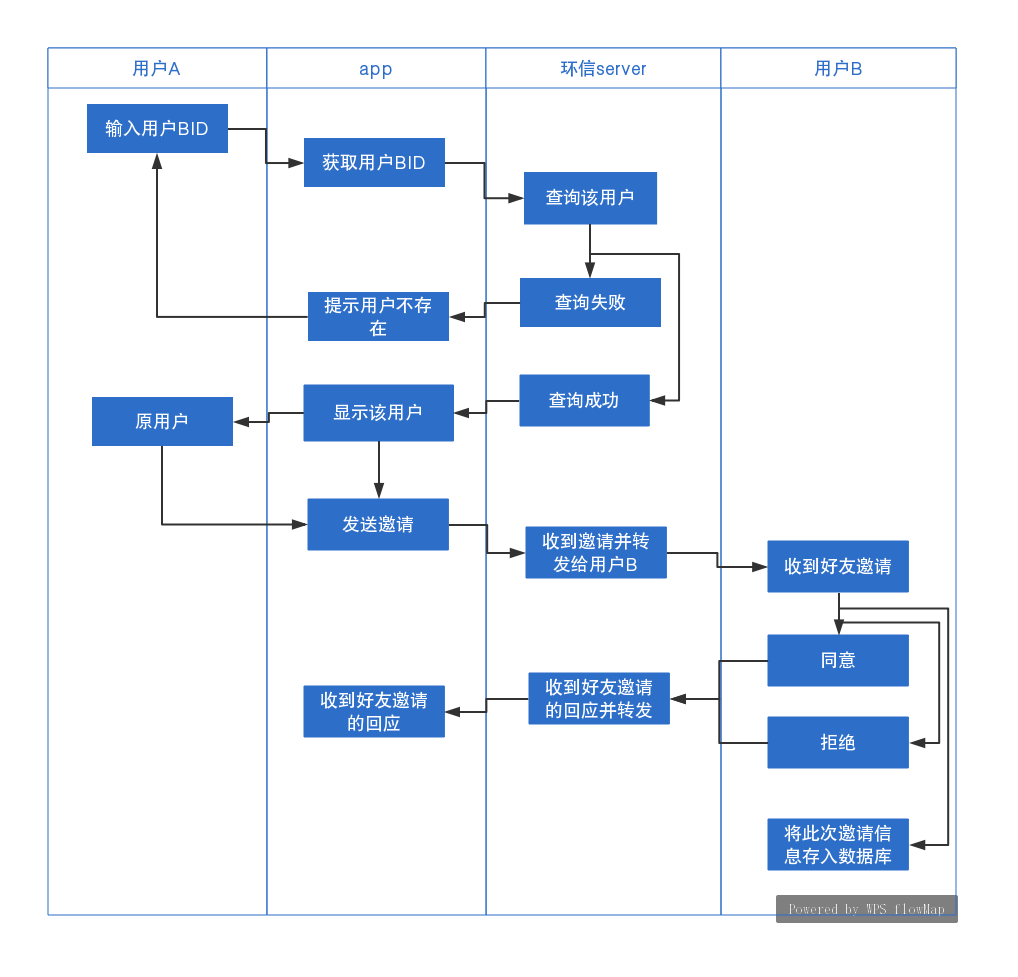


图3.3-3 添加好友流程图

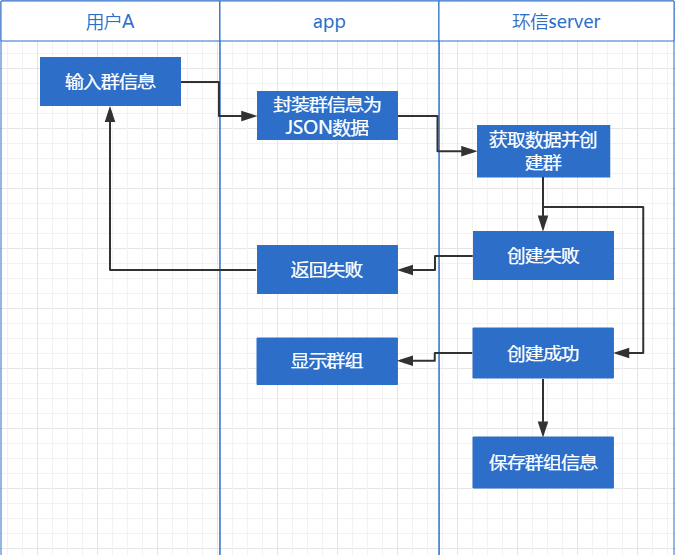


图3.3-4 创建群组流程图

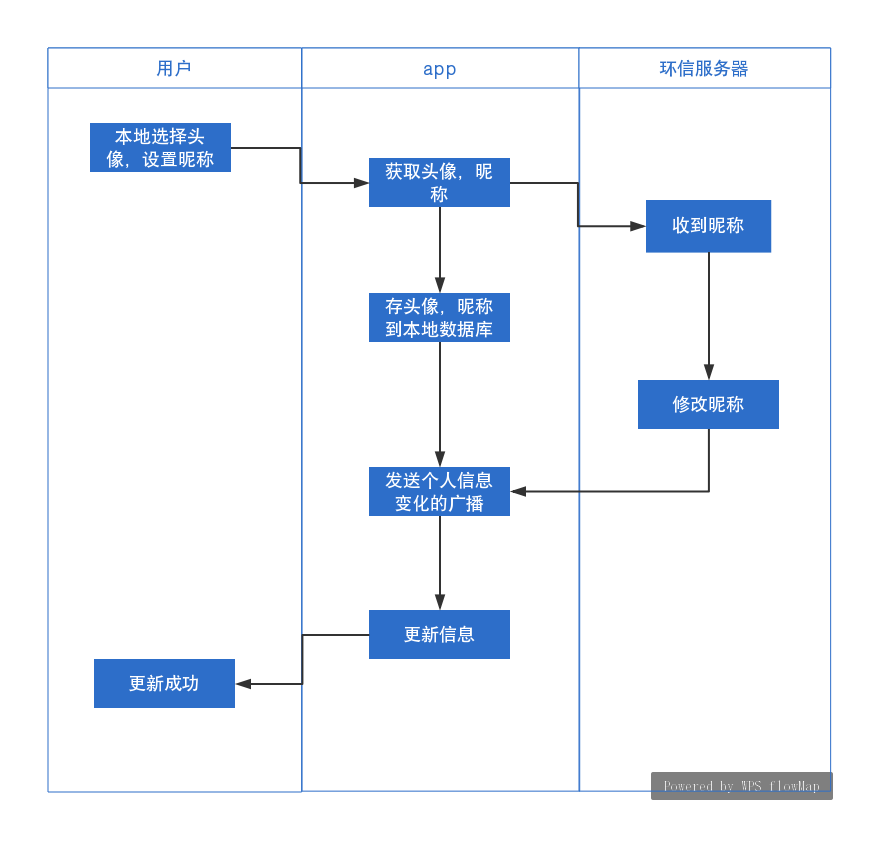


图3.3-5 修改头像和昵称流程图

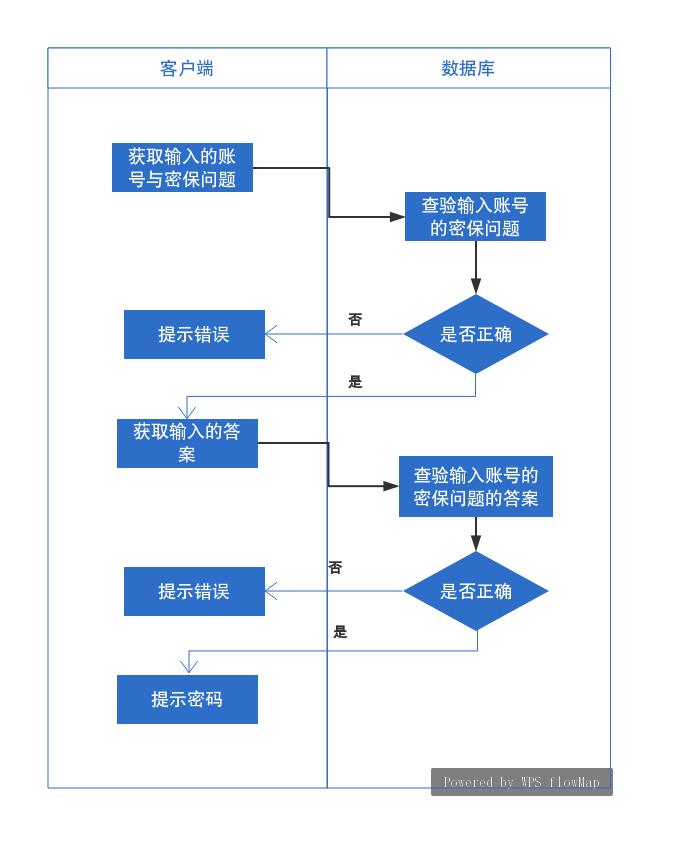


图3.3-6 找回密码流程图

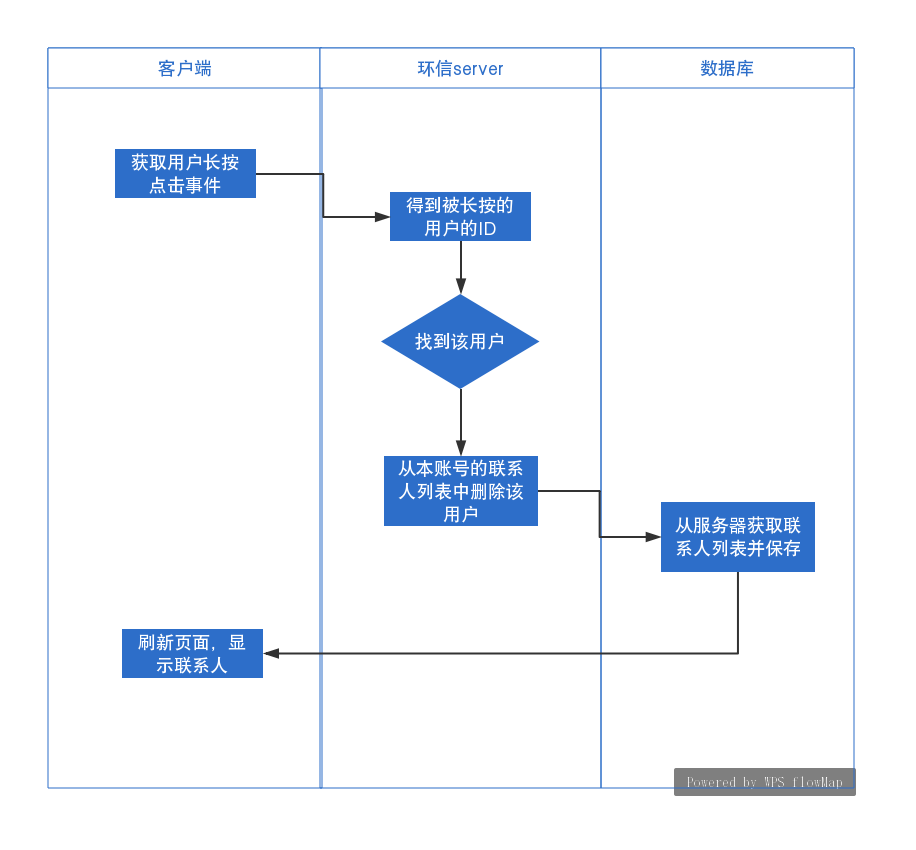


图3.3-7 删除好友流程图

## 3.4 外部接口

需调用Android的打开外部储存接口，设置头像时，需打开存储控件选择头像。

需调用摄像头接口，设置头像时与扫描二维码时，需打开摄像头

链接数据库使用jdbc接口

## 3.5 内部接口

登录调用环信SDK的登录接口，发送消息调用环信SDK内部接口注册，创建群组，添加好友，修改昵称，修改头像调用环信服务器的rest api接口。

# **4 数据库设计**

## 4.1 概念结构设计

### 4.1.1 设计思路

在一个聊天系统中，最重要的就是用户，因此有一个用户信息的数据库表，该表存储有每一个在次系统注册的用户，关于每一个用户的信息，都会存储到这张表里。此外，在每个用户的本地数据库还会生成一张用户的表，只存有本用户的相关信息，在用户的本地，还会生成一张联系人的表，存储有用户联系人信息。此外还有一个管理邀请信息的表和联系人信息的表。

1、各个实体的属性：

（1）用户信息：用户ID，用户姓名，用户昵称，用户头像，用户密码，密保问题及密保问题答案

（2）群信息：群名称，群ID，群的创建人，群简介，群的邀请限制，群的人数限制，

（3）邀请信息：邀请信息ID，发送邀请的用户ID，接收邀请的用户ID，是否为群邀请信息，群ID，邀请原因，邀请状态

（4）选择联系人信息：用户信息，是否被选中

2、各属性的类型：

（1）用户信息：用户ID、用户姓名、用户昵称、用户密码、密保问题，及密保问题答案均为text类型，用户头像为blob类型。

（2）群信息：群名称、群ID、群的创建人、群简介均为text类型，群的邀请限制和人数限制为int型。

（3）邀请信息：邀请信息ID、发送邀请的用户ID、接收邀请的用户ID、邀请原因为text类型，是否为群邀请信息以及邀请状态为int类型。

（4）选择联系人信息：用户信息为用户类型，是否被选中为boolean类型。

3、实体类的关系：

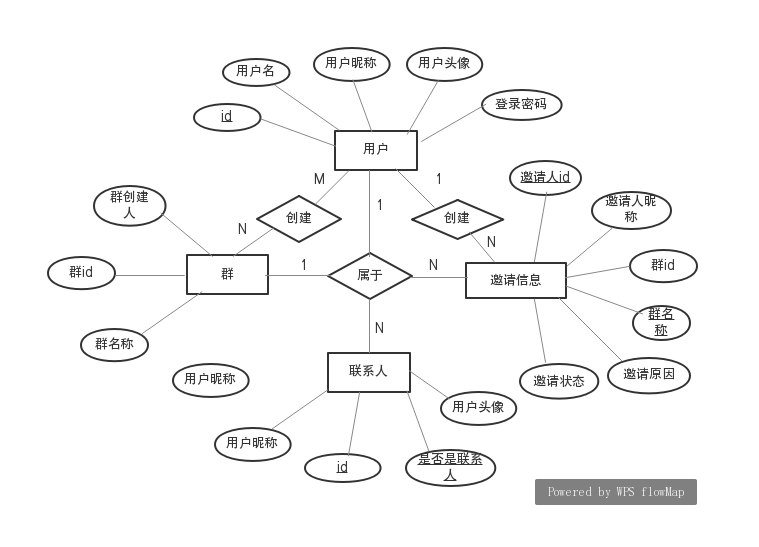
（1）一个用户ID对应一个用户信息，一个用户信息对应一个用户ID

（2）一条邀请信息对应一个用户ID，一个用户ID对应一到多条邀请信息

（3）一条邀请信息对应一个群ID，一个群ID对应一到多条邀请信息

（4）一个群对应一个创建人，一个创建者对应一到多个群

### 4.1.2 E-R图



## 4.2 逻辑结构设计

### 4.2.1 设计思路

### 4.2.2 逻辑模型

E-R图转换为关系表

（1）用户和群是M：N的关系，即一个用户可创建多个群，一个群同样可以属于多个用户

用户各列：用户名，用户ID，用户昵称，用户头像，用户密码

1. 用户和邀请信息是1：N的关系，即一个用户可有多条邀请信息，但是一条邀请信息只能是一个用户的

邀请信息各列：用户ID，用户名，群ID，群名称，邀请原因，邀请状态

1. 群和邀请信息是1：N的关系，即一个群可对多名联系人发出邀请从而产生多条邀请信息，但一个邀请信息只能属于一个群组
2. 用户和联系人是M：N的关系，即一个用户可拥有多个联系人，同样一个联系人属于多个用户

联系人各列：联系人ID，联系人姓名，联系人昵称，联系人头像，是否是联系人

## 4.3 物理结构设计

### 4.3.1 存取方式

数据库的存取方式采用索引法

### 4.3.2 存储结构

表4-1用户表

|  |  |  |  |
| --- | --- | --- | --- |
| 属性名 | 数据类型 | 备注 |  |
| id | text | 用户ID | 主键 |
| name | text | 姓名 |  |
| nick | text | 用户昵称 |  |
| picture | blob | 用户头像 |  |
| password | text | 用户密码 |  |
| question | text | 密保问题 |  |
| answer | text | 密保问题答案 |  |

表4-2联系人表

|  |  |  |  |
| --- | --- | --- | --- |
| 属性名 | 数据类型 | 备注 |  |
| id | text | 联系人ID | 主键 |
| name | text | 联系人姓名 |  |
| nick | text | 联系人昵称 |  |
| picture | blob | 联系人头像 |  |
| is\_contact | integer | 与我是否是联系人 |  |

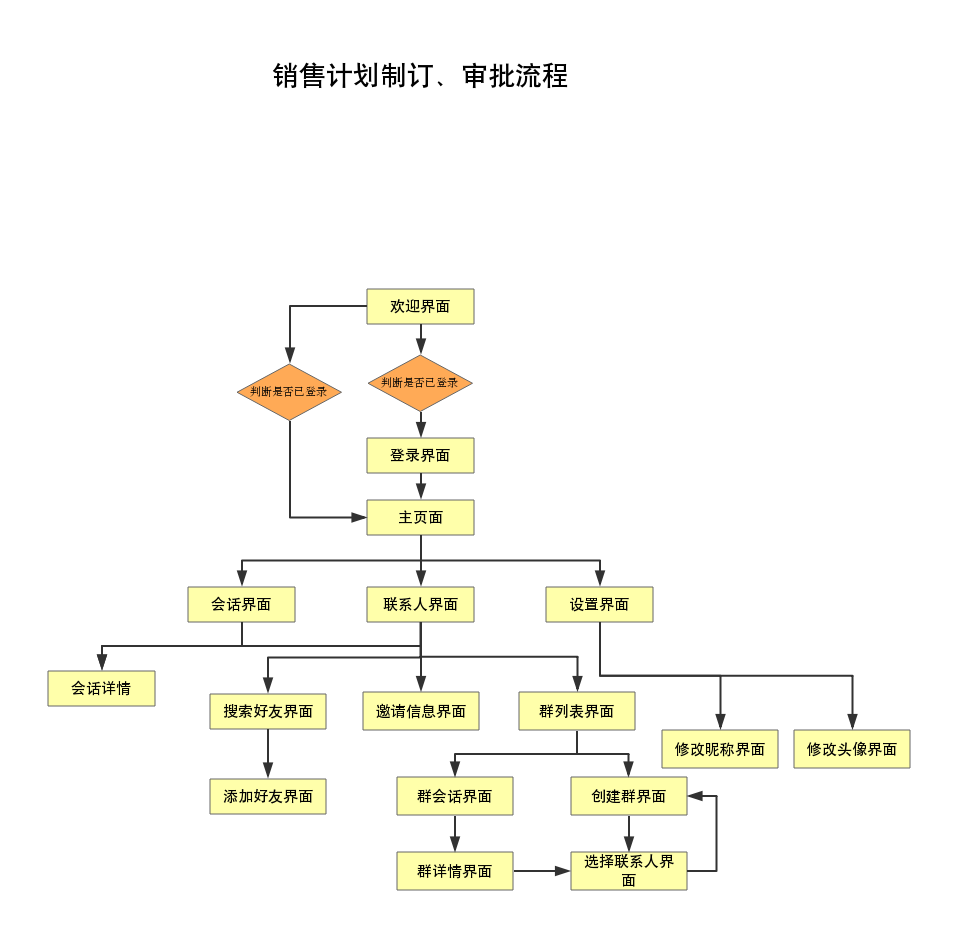
表4-3邀请信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 属性名 | 数据类型 | 备注 |  |
| fromuser\_id | text | 邀请人ID |  |
| touser\_id | text | 被邀请人ID |  |
| group\_id | text | 群ID |  |
| reason | text | 邀请原因 |  |
| status | int | 邀请状态 |  |
| id | text | 邀请信息ID | 主键 |
| is\_group | int | 是否为群邀请 |  |

5 界面设计

## 5.1 界面关系图或工作流图

首先，打开APP以后，入眼的是一张图片，我称它为欢迎界面，这个界面运行两秒以后会进入主页面或者是登录页面，如果上次已登录且没有退出登录，则直接进入主页面；如果没有登录，则进入登陆页面进行登录操作。登录以后进入主页面，有三个界面可选择操作，会话界面，联系人界面，设置界面。从会话界面可进入会话详情界面；联系人界面可进入搜索联系人界面，邀请信息界面，群列表界面，以及会话详情界面，设置界面可进入设置昵称界面和设置头像界面；从搜索联系人界面可进入添加联系人界面；从群列表界面可进入群会话界面以及创建群界面，从群会话界面和创建群界面均可进入选择联系人加群的界面。



## 5.2 界面设计成果

### 5.2.1 主界面

主页面在布局上遵守简洁大方，方便操作的原则。整体采用垂直布局，顶部为一个标题栏，往下为一个FrameLayout，是三个fragment的盛放布局，最下面是三个权重相同的按钮，用来切换fragment。三个fragment分别为会话fragment，联系人fragment，设置fragment。

会话fragment继承自环信的会话fragment，

联系人fragment继承自环信的联系人fragment，再次基础上，添加一个headview，即好友邀请与群组的view；headview之下便是联系人列表，采用listview来展示联系人列表。

设置fragment除一个标题栏的顶部布局，其下则为三个按钮，分别为修改昵称，修改头像和退出登录按钮，分别执行三种不同的操作。

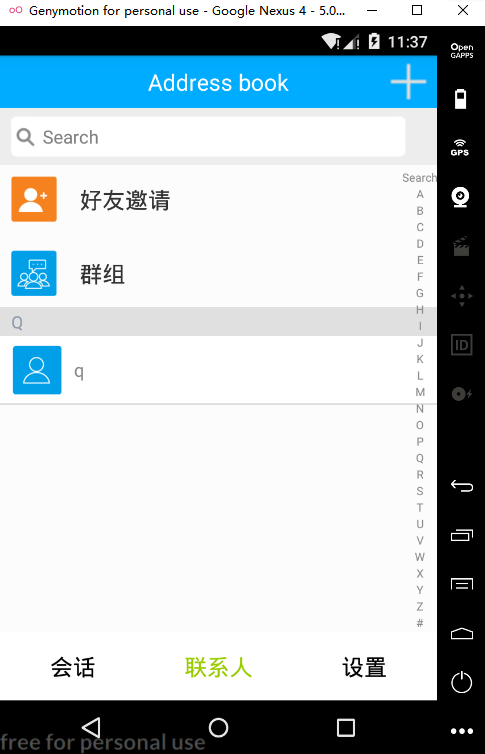


图5-1 主页面—联系人页面

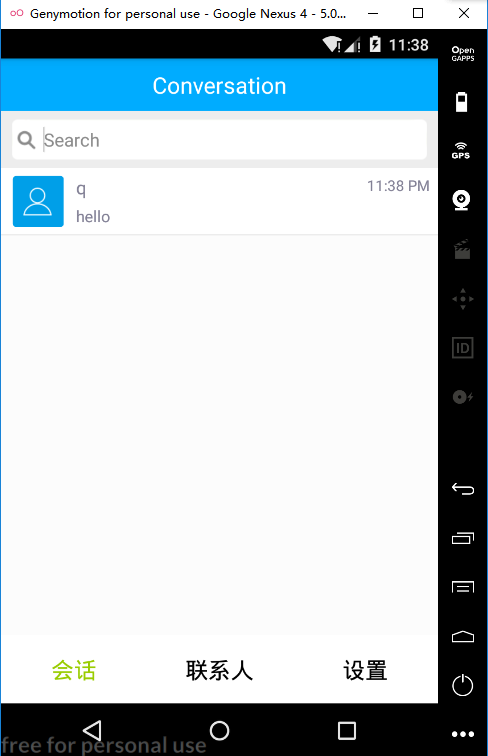


图5-2 主页面—会话界面图



图5-3 主页面—设置界面图

### 5.2.2 子界面

（1）欢迎页面

欢迎页面没有特殊布局，仅有一张图片，延迟两面进入APP



图5-4 子页面—欢迎页面图

1. 登录页面

登录页面总体布局为线性垂直布局，背景设为一张云空图，上方正中央为一个IamgeView，用来显示图片；其下方为两个线性水平布局，布局中仅有两个控件，左边为TextView，右边为EditView，用来显示账号和密码；在输入密码框的左下方有一选择框，为显示密码选择框，若勾选，则密码以明文显示，若不勾选，则以密文显示；勾选框下面则是登录按钮；位于页面底部的两个按钮分别是注册按钮和找回密码按钮，点击分别进入相应页面。

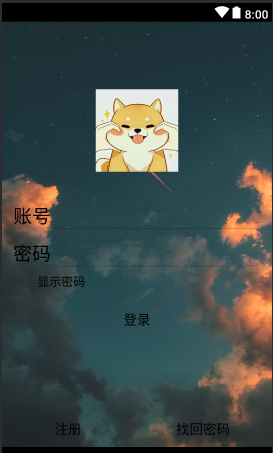


图5-5 子页面—登录页面图

1. 注册页面

注册页面整体布局与登录页面相似，为了提高用户的使用体验，背景设为云空图，中上部以黑色大字标明此为注册页面，其下方为两个水平线性布局，分别为注册的账号和密码，再往下则为取消和注册按钮。整体布局简洁大方，易于操作，为了提高用户的交互体验，提高用户的使用方便性，当注册成功与失败时，均会在页面下方弹出一个提示信息。



图5-6 子页面—注册页面图

1. 找回密码页面

找回密码页面整体布局与注册页面相似，为了提高用户的使用体验，背景设为云空图，中上部以黑色大字标明此为找回密码页面，其下方为三个水平线性布局，分别为找回密码的账号和密保问题还有用于显示密码的textview，再往下则为取消和注册按钮。整体布局简洁大方，易于操作，为了提高用户的交互体验，提高用户的使用方便性，当注册成功与失败时，均会在页面下方弹出一个提示信息。



图5-7 找回密码页面图

1. 搜索好友页面

搜索好友页面整体同样为垂直线性布局，最上方的标题栏标明添加联系人，右侧一个textview，并加以点击事件响应查找动作，标题栏下方为一个书写框，用于填写查找的用户id。EditText下面为一个隐藏布局，该布局用于显示查找的好友，当查找到该用户时，该布局可见，并且用户的头像和昵称都将显示，并且该布局设有点击响应事件，点击后进入好友详情页面。

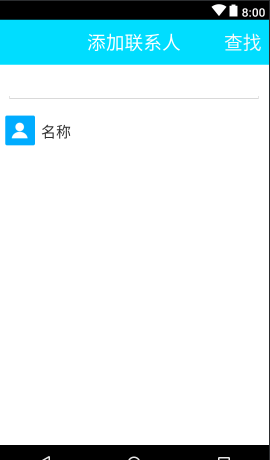


图5-8 搜索好友页面图

1. 好友详情页面

好友详情页面总体为一个垂直线性布局，总布局可分为三部分，上面为一个水平线性布局的用户头像、id、和昵称的小布局，中间为填写邀请原因的一个edittext，最下面为两个按钮的水平线性布局。

最上面的布局左侧为一个ImageView，用于显示用户头像，右侧为一个垂直线性布局，分别是用户ID和用户昵称。为了美观，三个小布局均与边界有一定的距离。

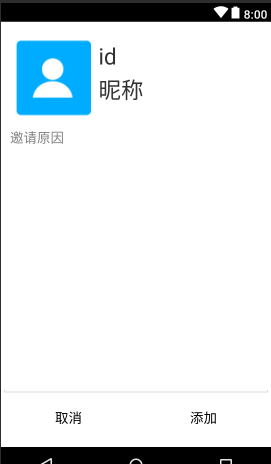


图5-9 好友详情页面图

1. 创建群页面

新建群组页面较为复杂，控件较多，但是布局起来也没有很难。总布局还是为垂直线性布局，最上面的titlebar标明为新建群组页面，往下为两个edittext，分别为群组名称和群组简介，下面是人数上限的编辑框，再往下为是否公开的checkbox和是否开放群邀请的checkbox，最下面就是一个button，用来响应点击事件，点击后跳转选择联系人进群聊的页面。



图5-10 新建群组页面图

1. 修改昵称和修改密码页面

修改昵称与修改密码布局相同，均为垂直线性布局，自上而下分别为一个TextView标明功能，其下为一个水平线性布局用于填写昵称或者密码，再往下为一个button来实现点击事件，获取昵称和密码分别修改环信sever和数据库中的昵称和密码。



图5-11 修改密码页面图

# **6详细设计**

### 6.1系统主要功能模块介绍

系统主要模块有注册，实现注册账号功能；登录，实现登录app功能；找回密码，实现找回密码功能；聊天，添加好友，实现搜索好友并添加功能；创建群聊，实现选择联系人创建群聊功能；修改个人信息，实现修改昵称，头像，密码及密保问题功能。

### 6.2 注册模块设计

6.2.1 注册模块算法描述

由于此聊天软件实在环信sdk的基础上开发的，所以服务器也是环信的服务器，而欢欣的服务器在集成用户体系时，对于用户信息只会保存用户的id，密码以及昵称，所以在注册时我们只需要将用户id和密码传到环信服务器即可。首先获取用户输入的id和密码，以及选择的密保问题和答案，将id与密码通过环信服务器的REST API接口发送到环信服务器，当成功收到返回值后，将用户id，密码，密保问题以及答案存入本地数据库中。

6.2.2 注册模块程序流程图

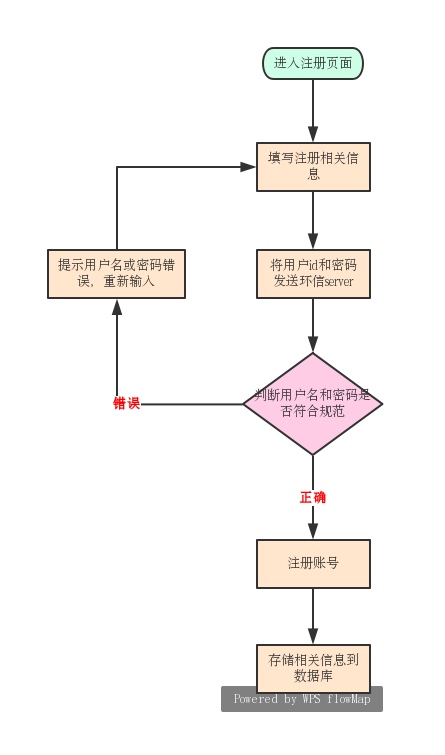


图6-1注册模块流程图

6.2.3 注册模块关键类说明

注册模块的关键类有两个，一为网络操作的工具类，类中封装了各个需要网络数据操作的方法，如注册， 添加好友等与环信服务器交互方法。二为注册Activity，其中实现了注册模块的基本逻辑和页面布局。

### 6.3 找回密码模块设计

6.3.1 找回密码算法描述

由于此系统没有个人服务器，所以用户的重要信息如密保问题及问题大难和密码只保存在本地，也只有当前用户在所有信息输入正确的时候，才能获取密码。具体过程是：首先获取用户输入的环信id以及选择的密保问题，通过环信id查询本地数据库所保存的密保问题，如果与现在选择的密保问题相同，则继续查验用户输入的密保问题答案与数据库中所保存的密保问题答案是否相同，若相同，则把用户密码显示在屏幕，若不相同，则提示密保问题答案错误。

6.3.2 找回密码程序流程图

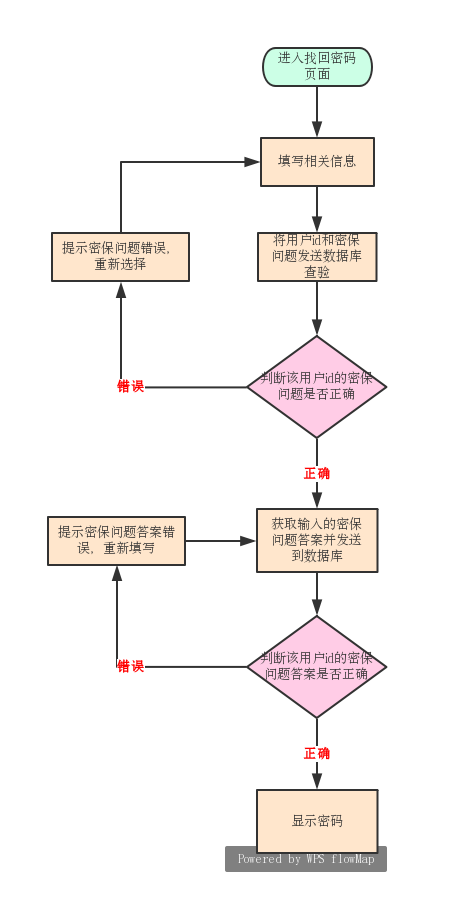


图6-2找回密码模块流程图

6.3.3 找回密码关键类说明

找回密码模块的关键类有两个，一为与数据库交互的工具类，类中封装了找回密码所需要的向数据库请求数据的方法。二为找回密码Activity，其中实现了找回密码模块的基本逻辑和页面布局。

### 6.4 聊天模块设计

6.4.1 聊天算法描述

相比其他模块，聊天模块是最重要的，而这模块的具体算法，环信已经帮我们封装好了。当我们发送消息时，需要将消息内容和接收者的id一起传给环信服务器，环信服务器进而发送给接收者。

6.4.2 聊天程序流程图

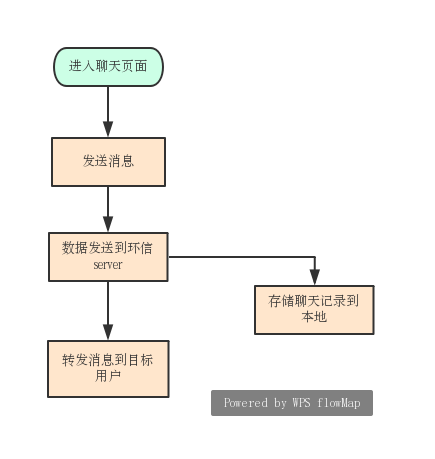


图63聊天模块流程图

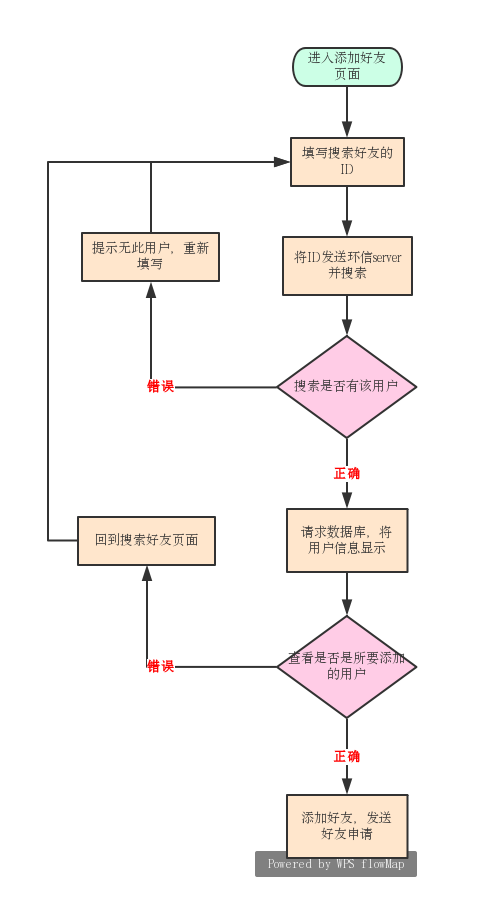
6.4.3 聊天关键类说明

### 6.5 添加好友模块设计

6.5.1 添加好友算法描述

打开添加好友页面后，当用户输入所要添加的联系人id后，会显示联系人，点击后进入联系人的详细页面，查看后在选择是否添加，确认添加以后，系统会调用环信服务器的REST API接口，将本用户id和要添加的用户id传给服务器，当服务器给两个用户确认聊天关系以后，通过返回的数据确认已添加成功，则将用户的相关信息存入数据库。

6.5.2 添加好友程序流程图



6.5.3 添加好友关键类说明

添加好友模块的关键类有三个，一为与数据库交互的工具类，类中封装了添加好友所需要的向数据库请求数据的方法。二为找回密码Activity，其中实现了添加好友模块的基本逻辑和页面布局。三为网络操作的工具类，类中封装了各个需要网络数据操作的方法，如注册， 添加好友等与环信服务器交互方法

### 6.6 创建群聊模块设计

6.6.1 创建群聊算法描述

点击创建群聊后，会进入创建群聊的页面，在此页面，用户输入创建群聊的相关信息，系统会将这些数据封装为一个json数据，并通过REST API传到服务器，服务器拿到这些书后，在服务器端创建一个群。创建号群后，系统通过网络请求获取群组信息后，显示在群列表中。

6.6.2 创建群聊程序流程图

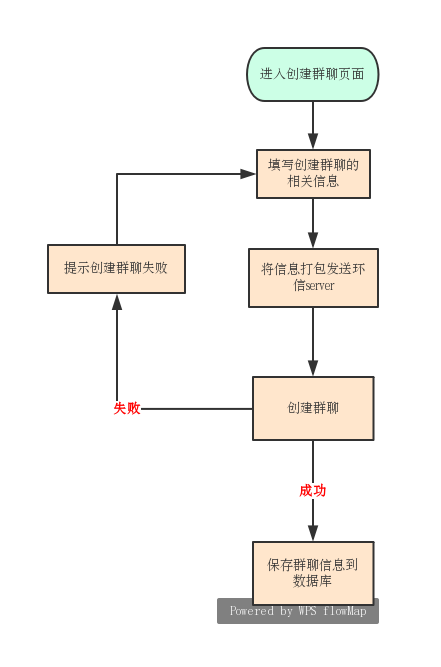


图6-5创建群聊模块流程图

6.6.3 创建群聊关键类说明

创建群聊模块的关键类有三个，一为与数据库交互的工具类，类中封装了创建群聊所需要的向数据库请求数据的方法。二为创建群聊Activity，其中实现了创建群聊模块的基本逻辑和页面布局。三为网络操作的工具类，类中封装了各个需要网络数据操作的方法，如注册， 添加好友等与环信服务器交互方法

### 6.7 修改昵称和密码模块设计

6.7.1 修改昵称和头像算法描述

进入修改昵称页面以后，用户输入想要修改为的昵称，系统调用相关方法获取当前用户的id，获取所输入的昵称，并将其发送给环信服务器，在环信服务器端修改用户昵称，之后在数据库中，将用户昵称修改。相比修改昵称，修改头像则简单许多，由于环信服务器的用户体系中只有环信id，密码以及用户昵称，所以用户的头像信息是不存储在环信服务器的，将其存储到数据库中，因此修改密码仅需要将其数据库中的头像信息修改即可。

6.7.2 修改昵称和密码程序流程图

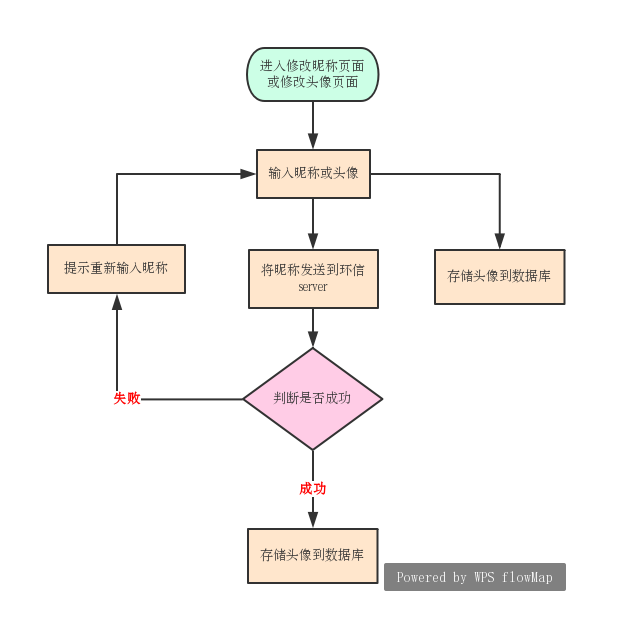


图6-6 修改昵称或密码模块流程图

6.7.3 修改昵称和密码关键类说明

修改昵称模块的关键类有三个，一为与数据库交互的工具类，类中封装了修改昵称所需要的向数据库请求数据的方法。二为修改昵称Activity，其中实现了修改昵称模块的基本逻辑和页面布局。三为网络操作的工具类，类中封装了各个需要网络数据操作的方法，如注册， 添加好友等与环信服务器交互方法

7 编码

## 7.1 代码实现与核心算法

与服务器通过REST API 实现交互

public class OKHttpUtils {  
  
 public static void getToken (okhttp3.Callback callback) {  
 String address = "https://a1.easemob.com/1108190122085684/mahaifeng-im/token";  
 String gson = "{\"grant\_type\":\"client\_credentials\",\"client\_id\":\"YXA6VkZncB-aEemZ5vO6rLH8Kw\",\"client\_secret\":\"YXA6rKqsDCY7gbMwKt-QJ2ZZqWWD5fs\"}";  
 OkHttpClient client = new OkHttpClient();  
 RequestBody requestBody = RequestBody.*create*(MediaType.*parse*("application/json; charset=utf-8"), gson);  
 Request request = new Request.Builder()  
 .header("content-type","application/json")  
 .post(requestBody)  
 .url(address)  
 .build();  
 client.newCall(request).enqueue(callback);  
 }  
  
 public static void rePassWord(String address, String password, String access\_token, okhttp3.Callback callback){  
 OkHttpClient okHttpClient = new OkHttpClient();  
 RequestBody requestBody = RequestBody.*create*(MediaType.*parse*("application/json; charset=utf-8"),"newpassword "+password);  
 Request request = new Request.Builder()  
 .addHeader("Authorization","Bearer "+access\_token)  
 .url(address)  
 .put(requestBody)  
 .build();  
 okHttpClient.newCall(request).enqueue(callback);  
 }  
  
 public static void reNick(String address,String nick,String access\_token,okhttp3.Callback callback) {  
 OkHttpClient client = new OkHttpClient();  
 RequestBody requestBody = RequestBody.*create*(MediaType.*parse*("application/json; charset=utf-8"),"nickname "+nick);  
 Request request = new Request.Builder()  
 .addHeader("Authorization","Bearer "+access\_token)  
 .url(address)  
 .put(requestBody)  
 .build();  
 client.newCall(request).enqueue(callback);  
 }  
  
 public static void registerNew(String address,String username,String password,String access\_token,okhttp3.Callback callback) {  
 OkHttpClient client = new OkHttpClient();  
 String post = "{\"username\":\""+username+"\",\"password\":\""+password+"\"}";  
 RequestBody requestBody = RequestBody.*create*(MediaType.*parse*("application/json; charset=utf-8"), post);  
 Request request = new Request.Builder()  
 .addHeader("Authorization","Bearer "+access\_token)  
 .url(address)  
 .post(requestBody)  
 .build();  
 client.newCall(request).enqueue(callback);  
 }  
  
 public static void addContact(String address,String access\_token,okhttp3.Callback callback){  
 OkHttpClient client = new OkHttpClient();  
 Request request = new Request.Builder()  
 .addHeader("Authorization","Bearer "+access\_token)  
 .url(address)  
 .post(null)  
 .build();  
 client.newCall(request).enqueue(callback);  
 }  
  
 public static void apiCreatGroup(String address, JSONObject jsonObject, String access\_token, okhttp3.Callback callback) {  
 OkHttpClient client = new OkHttpClient();  
 RequestBody requestBody = RequestBody.*create*(MediaType.*parse*("application/json; charset=utf-8"), jsonObject.toString());  
 Request request = new Request.Builder()  
 .addHeader("Authorization","Bearer "+access\_token)  
 .url(address)  
 .post(requestBody)  
 .build();  
 client.newCall(request).enqueue(callback);  
 }  
}

与数据库使用sql语句实现增删改查

public class UserAccountTableDao {  
  
 private final UserAccount userAccount;  
  
 public UserAccountTableDao(Context context) {  
 userAccount = new UserAccount(context);  
  
 }  
 //添加用户  
 public void addAccount (UserInfo userInfo){  
 //获取数据库  
 SQLiteDatabase database = userAccount.getReadableDatabase();  
 //封装用户信息  
 ContentValues values = new ContentValues();  
 values.put(UserAccountTable.*NAME*,userInfo.getName());  
 values.put(UserAccountTable.*ID*,userInfo.getUserId());  
 values.put(UserAccountTable.*PICTURE*,userInfo.getPicture());  
 values.put(UserAccountTable.*NICK*,userInfo.getNick());  
 values.put(UserAccountTable.*PASSWORD*,userInfo.getPassword());  
  
 database.replace(UserAccountTable.*TABLE\_NAME*,null,values);  
 }  
 //获取用户信息  
 public UserInfo getAccountInfo(String id){  
 //获取数据库  
 SQLiteDatabase database = userAccount.getReadableDatabase();  
 //查询  
 String sql = "select \* from " + UserAccountTable.*TABLE\_NAME* + " where " + UserAccountTable.*ID* + " =? ";  
 Cursor cursor = database.rawQuery(sql, new String[]{id});  
 UserInfo userInfo = null;  
 if (cursor.moveToNext()) {  
 userInfo = new UserInfo();  
 //封装  
 userInfo.setName(cursor.getString(cursor.getColumnIndex(UserAccountTable.*NAME*)));  
 userInfo.setUserId(cursor.getString(cursor.getColumnIndex(UserAccountTable.*ID*)));  
 userInfo.setNick(cursor.getString(cursor.getColumnIndex(UserAccountTable.*NICK*)));  
 userInfo.setPicture(cursor.getBlob(cursor.getColumnIndex(UserAccountTable.*PICTURE*)));  
 userInfo.setPassword(cursor.getString(cursor.getColumnIndex(UserAccountTable.*PASSWORD*)));  
 }  
 //关闭  
 cursor.close();  
 return userInfo;  
 }  
  
 /\*public void updateAccount(UserInfo userInfo){  
  
 }\*/  
  
 //根据ID获取数据库中用户头像并返回为二进制字节数组  
 public byte[] getPictureFromSql(String id) {  
 byte[] blob = null;  
 SQLiteDatabase database = userAccount.getReadableDatabase();  
 String sql = " select \* from " + UserAccountTable.*TABLE\_NAME* + " where " + UserAccountTable.*ID* + " =?";  
 Cursor cursor = database.rawQuery(sql, new String[]{id});  
 while (cursor.moveToNext()){  
 blob = cursor.getBlob(cursor.getColumnIndexOrThrow(UserAccountTable.*PICTURE*));  
 }  
 cursor.close();  
 return blob;  
 }  
  
 //将得到的图片转为二进制存到数据库  
 public long setPictureToSql(String id,Bitmap bitmap) {  
 if (bitmap == null){  
 return -1;  
 }  
 ByteArrayOutputStream byteArrayOutputStream = new ByteArrayOutputStream();  
 bitmap.compress(Bitmap.CompressFormat.*PNG*, 100, byteArrayOutputStream);  
 SQLiteDatabase database = userAccount.getReadableDatabase();  
 ContentValues contentValues = new ContentValues();  
 contentValues.put(UserAccountTable.*PICTURE*,byteArrayOutputStream.toByteArray());  
 long replace = database.update(UserAccountTable.*TABLE\_NAME*,contentValues,UserAccountTable.*ID* + " =?",new String[]{id});  
 return replace;  
 }  
  
 public void reNick(String id, String nick){  
 SQLiteDatabase database = userAccount.getReadableDatabase();  
 ContentValues value = new ContentValues();  
 value.put(UserAccountTable.*NICK*,nick);  
 database.update(UserAccountTable.*TABLE\_NAME*,value,UserAccountTable.*ID* + " =? ",new String[]{id});  
 }  
  
 public String getPassWord(String id){  
 SQLiteDatabase database = userAccount.getReadableDatabase();  
 String sql = " select \* from " + UserAccountTable.*TABLE\_NAME* + " where " + UserAccountTable.*ID* + " =?";  
 Cursor cursor = database.rawQuery(sql, new String[]{id});  
 String string = cursor.getString(cursor.getColumnIndex(UserAccountTable.*PASSWORD*));  
 return string;  
 }  
}

## 7.2 代码优化分析

本系统是Android聊天软件，主要的算法就是对数据库的存取和与后台服务器的交互。需要做的优化为sql查询语句的优化。而本系统所做的查询是简单的单表查询，所以优化方面侧重不多。

8 测试

## 8.1 测试方案设计

### 8.1.1 测试策略

对本系统的各项功能和界面UI进行人工测试，UI测试主要包括，用户界面的色彩搭配是否合理，整体布局是否正确，样式是否统一。功能测试主要包括，各项功能的实现情况，是否有使用bug，是否符合需求。

### 8.1.2 测试进度安排

先进行界面UI测试，通过之后进行功能测试。

### 8.1.3 测试资源

1. 测试环境

软件环境：开发软件为Android studio，运行软件为genymotion，操作系统为windows10.

硬件环境：客户端为自己的电脑，网络为局域网。

1. 测试工具

测试工具为Android studio debug。

1. 测试人员

测试人员为马海峰。

### 8.1.4 关键测试点

关键测试点为注册账号、添加好友、创建群组、聊天和修改个人信息。

## 8.2 测试用例构建

### 8.2.1 测试用例编写约定

编写原则：基本原则为：“一点多例”。测试用例具有可测性，可验证性，全面性。

### 8.2.2 测试用例设计

测试用例：首先注册两个账号，让这两个账号相互加为好友，然后创建群组，两人相互聊天，群聊，然后修改昵称，修改头像，修改密码和密保。

### 8.2.3 关键测试用例

两人相互聊天，群聊，然后修改昵称，修改头像，修改密码和密保。

### 8.2.4 测试用例维护

在测试过程中发现测试不足，及时更新测试用例。

9 总结与展望

## 9.1 设计工作总结

此次毕业设计对于我来说有着深远的意义，Android编程属于我自学的，从来没有独立完成过一个完整的项目，可以说这是我第一次完整的完成一个app，这个app也是真正意义上的我的第一个app，虽说实现这个项目过程中用到了第三方的开发工具包和第三方的服务器，但尽管如此，我的能力还是得到了很大的锻炼，甚至通过这个项目，我找到了我的第一份工作。

项目的实现并不是一帆风顺的。过程中我也遇到了各种各样的问题，最让我印象深刻的是一个特别简单的问题：最开始在本地建立第一个数据存储表示，有一个参数写错了，但是当时没有发现，就继续写下去，但是运行查询数据时，返回的总是数据不存在。回过头去排查时，发现建表语句写错了，但是当我把建表语句改过后，重新运行程序，依然查询不到数据，这下我就彻底不不知所措了，该改的地方都已经改过，百思不得其解，就这样这个bug困扰我几天以后，偶然间查到，当本地建好表以后，修改建表语句不会重新建表，需要将以前的表删掉，才会重新建表，恍然大悟。就这样删掉前面的表以后，一切都回归正轨了。虽然这是个非常小的问题，却是最让我印象深刻的。

通过这次项目，我还学习到了Android线程切换，网络数据请求，线程间的通信等相关知识，更是扎实了我的基本功。在项目过程中，好多只是都是随学随用，比如debug，以前只是写些小的demo，也用不到debug模式，通过这次项目，也是熟悉了如何运用debug去寻找问题的所在。还有各种环境的配置问题，繁琐数据解析问题等。最难的还是修改bug，有句话现在我深的体会：代码2分钟，bug两小时。修改bug占据了系统开发的大部分时间，但在老师，同学，网络和自己的努力下，不怕麻烦，不怕重复，每当克服了困难，修改了bug，收获的不仅仅是一个运行良好的系统，更是一个作为程序员的喜悦，同时也感觉到自己的知识，在不断地积累，在不知不觉中增加。

## 9.2 未来工作展望

相比现在国内外即时通讯的研究现状，我的这个系统还显得不成熟，还存在着诸多的问题，例如没有自己独立的服务器，信息没有加密，数据存储不够安全，功能不够完善等。在后续的工作中，如果有时间，我会继续完成这个项目，使其更加完善，更加成熟。

谢 辞

首先我特别感谢在我这次毕业设计中给予我帮助的丁琼老师，她会耐心的为我制定设计计划，每个星期监督我完成本周的任务，认真修改我写的各项文档以及论文初稿终稿，标注出错误与不足的地方。再者我要感谢毕业设计这次机会，让我明白只要肯努力去学，我可以学会很多东西。最后我要感谢学校，培育了我四年。大学四年，我交到了很多朋友，也学会了多门计算机语言，是大学给予了我学习和成长的机会。

现在我即将毕业了，希望我能够快速的融入新的环境和生活。

参考文献

[1] 王珊，萨师煊. 数据库系统概论［M］. 高等教育出版社，2006：1~418.

[2] 李建中、王珊.《数据库系统原理（第2版）》[M]. 北京：电子工业出版社，2004-09：209-273.

[3] 黄缙华等. MySQL入门很简单[M].清华大学出版社，2011.01.

[4] 郭霖. Android第一行代码［M］.

[5]环信. 环信即时通讯云V3.0［EB/OL］.http://http://docs-im.easemob.com/im/start

[6]环信. 环信服务器端集成Swagger文档［EB/OL］ . http://api-docs.easemob.com/#!/%E5%8F%91%E9%80%81%E6%B6%88%E6%81%AF/post\_org\_name\_app\_name\_messages

附录A 外文翻译—原文部分

Wikipedia

Instant Messaging (IM) is a form of real-time communication between two or more people based on typed text. [The](https://www.baidu.com/s?wd=The&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank) text is conveyed via computers connected over a network such as [the](https://www.baidu.com/s?wd=the&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank) Internet.  
Overview  
Instant messaging (IM) and chat are technologies that create [the](https://www.baidu.com/s?wd=the&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank) possibility of real-time text-based communication between two or more participants over the internet or some form of internal network/ intranet. It is important to understand that what separates chat and instant messaging from technologies such as e-mail is the perceived synchronicity of the communication by the user - Chat happens in real-time before your eyes. Some systems allow the sending of messages to people not currently logged on (offline messages), thus removing much of the difference between Instant Messaging and e-mail.  
While many IM services have additional features such as: the immediate receipt of acknowledgment or reply, group chatting, conference services (including voice and video), conversation logging and file transfer, those functions are beyond the scope of this article.  
IM allows effective and efficient communication, featuring immediate receipt of acknowledgment or reply. In certain cases Instant Messaging involves additional features, which make it even more popular, i.e. to see the other party, e.g. by using web-cams, or to talk directly for free over the Internet.  
It is possible to save a conversation for later reference. Instant messages are typically logged in a local message history which closes the gap to the persistent nature of e-mails and facilitates quick exchange of information like URLs or document snippets (which can be unwieldy when communicated via telephone).  
History  
Instant messaging actually predates the Internet, first appearing on multi-user operating systems like CTSS and Multics[1] in the mid-1960s. Initially, many of these systems, such as CTSS'.SAVED, were used as notification systems for services like printing, but quickly were used to facilitate communication with other users logged in to the same machine. As networks developed, the protocols spread with the networks. Some of these used a [peer-to-peer](https://www.baidu.com/s?wd=peer-to-peer&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank) protocol (eg talk, ntalk and ytalk), while others required peers to connect to a server (see talker and IRC). During the Bulletin board system (BBS) phenomenon that peaked during the 1980s, some systems incorporated chat features which were similar to instant messaging; Freelancin' Roundtable was one prime example.  
In the last half of the 1980s and into the early [1990s](https://www.baidu.com/s?wd=1990s&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank), the Quantum Link online service for Commodore 64 computers offered user-to-user messages between currently connected customers which they called "On-Line Messages" (or OLM for short). Quantum Link's better known later incarnation, America Online, offers a similar product under the name "AOL Instant Messages" (AIM). While the Quantum Link service ran on a Commodore 64, using only the Commodore's PETSCII text-graphics, the screen was visually divided up into sections and OLMs would appear as a yellow bar saying "Message From:" and the name of the sender along with the message across the top of whatever the user was already doing, and presented a list of options for responding.[2] As such, it could be considered a sort of GUI, albeit much more primitive than the later Unix, Windows and Macintosh based GUI IM programs. OLMs were what Q-Link called "Plus Services" meaning they charged an extra per-minute fee on top of the monthly Q-Link access costs.  
Modern, Internet-wide, GUI-based messaging clients, as they are known today, began to take off in the mid [1990s](https://www.baidu.com/s?wd=1990s&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank) with ICQ (1996) being the first, followed by AOL Instant Messenger (AOL Instant Messenger, 1997). AOL later acquired Mirabilis, the creators of ICQ. A few years later ICQ (by now owned by AOL) was awarded two patents for instant messaging by the [U.S.](https://www.baidu.com/s?wd=U.S.&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank) patent office. Meanwhile, other companies developed their own applications (Excite, MSN, Ubique, and Yahoo), each with its own proprietary protocol and client; users therefore had to run multiple client applications if they wished to use more than one of these networks. In 1998 IBM released IBM Lotus Sametime, a product based on technology acquired when IBM bought Haifa-based Ubique and Lexington-based Databeam.  
In 2000, an open source application and open standards-based protocol called Jabber was launched. Jabber servers could act as gateways to other IM protocols, reducing the need to run multiple clients. Multi-protocol clients such as Digsby,Pidgin, Trillian, Adium and Miranda can use any of the popular IM protocols by using additional local libraries for each protocol. IBM Lotus Sametime's November 2007 release added IBM Lotus Sametime Gateway support for XMPP.  
Recently, many instant messaging services have begun to offer video conferencing features, Voice Over IP (VoIP) and web conferencing services. Web conferencing services integrate both video conferencing and instant messaging capabilities. Some newer instant messaging companies are offering desktop sharing, IP radio, and IPTV to the voice and video features.  
The term "instant messenger" is a service mark of Time Warner[3] and may not be used in software not affiliated with AOL in the United States. For this reason, the instant messaging client formerly known as Gaim or gaim announced in April 2007 that they would be renamed "Pidgin"  
Cooperation  
Standard free instant messaging applications offer functions like file transfer, contact lists, the ability to have simultaneous conversations etc. These may be all the functions that a small business needs but larger organisations will require more sophisticated applications that can work together. The solution to finding applications capable of this is to use enterprise versions of instant messaging applications. These include titles like Jabber, Lotus Sametime, Microsoft Office Communicator, etc., which are often integrated with other enterprise applications such as workflow systems. These enterprise applications, or Enterprise Application Integration (EAI), are built to certain constraints, namely storing data in a common format.  
There have been several attempts to create a unified standard for instant messaging: IETF's SIP (Session Initiation Protocol) and SIMPLE (SIP for Instant Messaging and Presence Leveraging Extensions), APEX (Application Exchange), Prim (Presence and Instant Messaging Protocol), the open XML-based XMPP (Extensible Messaging and Presence Protocol), more commonly known as Jabber and OMA's (Open Mobile Alliance) IMPS (Instant Messaging and Presence Service) created specifically for mobile devices.Most attempts at creating a unified standard for the major IM providers (AOL, [Yahoo!](https://www.baidu.com/s?wd=Yahoo!&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank) and Microsoft) have failed and each continues to use its own proprietary protocol.However, while discussions at IETF were stalled, Reuters head of collaboration services, David Gurle (the founder of Microsoft's Real Time Communication and Collaboration business), signed the first inter-service provider connectivity agreement on September 2003. This agreement enabled AIM, ICQ and MSN Messenger users to talk with Reuters Messaging counterparts and vice-versa against an access fee. Following this, Microsoft, [Yahoo!](https://www.baidu.com/s?wd=Yahoo!&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank) and AOL came to a deal where Microsoft's Live Communication Server 2005 users would also have the possibility to talk to public instant messaging users. This deal established SIP/SIMPLE as a standard for protocol interoperability and established a connectivity fee for accessing public instant messaging clouds. Separately, on October 13, 2005 Microsoft and [Yahoo!](https://www.baidu.com/s?wd=Yahoo!&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank) announced that by (the Northern Hemisphere) summer of 2006 they would interoperate using SIP/SIMPLE which is followed on December 2005 by the AOL and Google strategic partnership deal where Google Talk users would be able to talk with AIM and ICQ users provided they have an identity at AOL.  
There are two ways to combine the many disparate protocols:  
One way is to combine the many disparate protocols inside the IM client application.  
The other way is to combine the many disparate protocols inside the IM server application. This approach moves the task of communicating to the other services to the server. Clients need not know or care about other IM protocols. For example, LCS 2005 Public IM Connectivity. This approach is popular in Jabber/XMPP servers however the so-called transport projects suffer the same reverse engineering difficulties as any other project involved with closed protocols or formats.  
Some approaches, such as that adopted by the Sonork enterprise IM software or the Jabber/XMPP network or Winpopup LAN Messenger, allow organizations to create their own private instant messaging network by enabling them to limit access to the server (often with the IM network entirely behind their firewall) and administer user permissions. Other corporate messaging systems allow registered users to also connect from outside the corporation LAN, by using a secure firewall-friendly HTTPS based protocol. Typically, a dedicated corporate IM server has several advantages such as pre-populated contact lists, integrated authentication, and better security and privacy.  
Some networks have made changes to prevent them from being utilized by such multi-network IM clients. For example, Trillian had to release several revisions and patches to allow its users to access the MSN, AOL, and Yahoo! networks, after changes were made to these networks. The major IM providers typically cite the need for formal agreements as well as security concerns as reasons for making these changes.  
  
Mobile Instant Messaging  
Mobile Instant Messaging (MIM) is a presence enabled messaging service that aims to transpose the desktop messaging experience to the usage scenario of being on the move. While several of the core ideas of the desktop experience on one hand apply to a connected mobile device, others do not: Users usually only look at their phone's screen — presence status changes might occur under different circumstances as happens at the desktop, and several functional limits exist based on the fact that the vast majority of mobile communication devices are chosen by their users to fit into the palm of their hand. Some of the form factor and mobility related differences need to be taken into account in order to create a really adequate, powerful and yet convenient mobile experience: radio bandwidth, memory size, availability of media formats, keypad based input, screen output, CPU performance and battery power are core issues that desktop device users and even nomadic users with connected network.  
Friend-to-friend networks  
Instant Messaging may be done in a Friend-to-friend network, in which each node connects to the friends on the friendslist. This allows for communication with friends of friends and for the building of chatrooms for instant messages with all friends on that network.  
Emotions are often expressed in shorthand. For example; lol. But a movement is currently underway to be more accurate with the emotional expression. Real time reactions such as (chortle) (snort) (guffaw) or (eye-roll) are rapidly taking the place of acronyms.[citation needed]  
Business application  
Instant messaging has proven to be similar to personal computers, e-mail, and the WWW, in that its adoption for use as a business communications medium was driven primarily by individual employees using consumer software at work, rather than by formal mandate or provisioning by corporate information technology departments. Tens of millions of the consumer IM accounts in use are being used for business purposes by employees of companies and other organizations.  
In response to the demand for business-grade IM and the need to ensure security and legal compliance, a new type of instant messaging, called "Enterprise Instant Messaging" ("EIM") was created when Lotus Software launched IBM Lotus Sametime in 1998. Microsoft followed suit shortly thereafter with Microsoft Exchange Instant Messaging, later created a new platform called Microsoft Office Live Communications Server, and released Office Communications Server 2007 in October 2007. Both IBM Lotus and Microsoft have introduced federation between their EIM systems and some of the public IM networks so that employees may use a single interface to both their internal EIM system and their contacts on AOL, MSN, and Yahoo!. Current leading EIM platforms include IBM Lotus Sametime, Microsoft Office Communications Server, and Jabber XCP. In addition, industry-focused EIM platforms such as IMtrader from Pivot Incorporated, Reuters Messaging, and Bloomberg Messaging provide enhanced IM capabilities to financial services companies.  
  
The adoption of IM across corporate networks outside of the control of IT organizations creates risks and liabilities for companies who do not effectively manage and support IM use. Companies implement specialized IM archiving and security products and services like those from Secure Computing, Akonix, SurfControl, and ScanSafe to mitigate these risks and provide safe, secure, productive instant messaging capabilities to their employees.  
Practical Use in Enterprise  
The popular embrace of IM technology for sharing information has quickly led to organizations adopting IM solutions for the perceived advantages that can be brought by it. As organizations are becoming more information based (McNurlin & Sprague, 2006, p.499) the need for effective knowledge sharing, team working and collaborative environments amongst employees has become vital, especially within more geographically dispersed teams.  
Typically IM conversations tend to have a certain "character", they are often short and only cover one topic. Media-switching and multitasking are common throughout, however IM might also be used between established coworkers and friends for longer, more intermittent conversation. In their report of IM use at the workplace Nardi et al. (2000) identifies the four primary functions of IM which are often cited in other reports, These primary functions are:  
Quick Questions and Clarifications  
Coordinating and scheduling tasks  
Coordinating impromptu social meetings  
Keeping in touch with friends and family  
IM is perhaps best suited to "Quick Questions and Clarifications" as this is the most often mentioned attribute in other reports. A user can "respond rapidly without the overhead of telephone or FTF interaction. For example, IDC reports, "Users see IM as a medium for quick, semi-permanent ‘flashes’ that beg a near-immediate response" (Isaacs et al., 2002). Nardi's second and third observations are enabled in part due to the "Presence Awareness" feature of IM clients in which the user knows who is "available". This is the most relevant for colleagues who share the same physical space as each other and even paves the way for other mediums to take up the task of communication e.g. F2F or Phone. The implication is that viable communication of any sort can in someway be encouraged through IM's "Presence Awareness" feature. (Issacs et al, 2002) supports this view, "IM in business might not be the main tool for of communication, it could just be the meeting point for another type of media e.g. conference calls.  
Nardi's third and fourth observations focus on the social use of IM, which have also been widely publicized in other report. That IM is used for keeping in touch with friends and arranging social events has led some employers to believe that it is used primarily for this purpose. According to (Issacs et al, 2002) a market study found that "'Fear of losing employee productivity’ was the greatest concern of businesses in regards to instant messaging".  
The study by (Issacs et al, 2002) goes on to suggest this fear is unfounded as it was found that on average "only 13% of conversations contained personal topics", and "only 6.4% were exclusively personal".  
A study published in 2007, in the Journal of Computer Mediated Communication, found that workers who used instant messaging on the job reported less interruption than colleagues who did not.[5] Research showed that instant messaging was often used as a substitute for other, more disruptive forms of communication such as the telephone, e-mail, and face-to-face conversations. They found that the conversations were briefer between co-workers when using instant messaging than with the other forms of communications. Because of its unique setup, instant messaging allows users to control how and when they communicate with their coworkers. This technology gives people the ability to flag their availability or postpone responses to a more convenient time, according to one of the researchers.[6]  
 Review of Products  
"IM solutions can typically be catagorised into two types: Enterprise Instant Messaging (EIM) and Consumer Instant Messaging (CIM). Enterprise solutions use an internal IM server, however this isn't always feasible, particularly for smaller businesses with limited budgets. The second option, using a (CIM) provides the advantage of being inexpensive to implement and has little need for investing in new hardware or server software. However, in recent years open source IM clients such as Jabber have emerged that provide free EIM grade solutions.

附录B 外文翻译—译文部分

即时消息(IM)是两个人或两个人以上基于键入文本的实时通信形式。文本通过连接在网络(如Internet)上的计算机进行传输。

概述

即时消息(IM)和聊天是一种技术，它使两个或多个参与者通过internet或某种形式的内部网络/内部网进行基于文本的实时通信成为可能。重要的是要明白，将聊天和即时消息与电子邮件等技术区分开来的是用户交流的同步性——聊天是在你眼前实时发生的。有些系统允许向当前未登录的用户(脱机消息)发送消息，从而消除了即时消息和电子邮件之间的许多差异。

虽然许多IM服务具有其他功能，例如:即时接收确认或回复、分组聊天、会议服务(包括语音和视频)、会话日志和文件传输，但这些功能超出了本文的讨论范围。

IM允许有效和高效的通信，以立即收到确认或回复为特点。在某些情况下，即时通讯包含了额外的功能，这使得它更受欢迎，例如通过网络摄像头看到对方，或者直接在互联网上免费交谈。

可以将对话保存起来供以后参考。即时消息通常记录在本地消息历史中，这弥补了与电子邮件持久性的差距，并促进了url或文档片段等信息的快速交换（当通过电话进行通信时，这些信息可能很笨拙）。

历史

实际上，即时消息比Internet早，最早出现在多用户操作系统上，如CTSS和Multics[1]，是在20世纪60年代中期。最初，许多这样的系统，如CTSS'。它被用作打印等服务的通知系统，但很快就被用于方便与登录到同一台机器的其他用户的通信。随着网络的发展，协议在网络中传播。其中一些使用对等协议(如talk、ntalk和ytalk)，而另一些则要求对等方连接到服务器(参见talker和IRC)。在1980年代达到高峰的BBS现象期间，一些系统加入了类似即时消息的聊天功能;自由职业者圆桌会议就是一个很好的例子。

在20世纪80年代末到90年代初，为Commodore 64计算机提供的量子链路在线服务在当前连接的客户之间提供用户对用户的消息，他们称之为“在线消息”(简称OLM)。量子链路后来的版本“美国在线”(America Online)更为人所知，它以“AOL即时消息”(AIM)的名义提供类似的产品。虽然量子链接服务运行在一个Commodore 64,只使用海军准将的PETSCII text-graphics,屏幕视觉分为部分,洞螈会出现一个黄色的酒吧说”的信息:“与消息一起发送方的名称在用户已经做的事情,并为反应提供了一个选项列表。因此，它可以被视为一种GUI，尽管它比后来基于Unix、Windows和Macintosh的GUI IM程序要原始得多。OLMs是Q-Link所称的“附加服务”，意思是他们在每月Q-Link访问成本的基础上，每分钟收取额外的费用。

现代的、internet范围的、基于gui的消息传递客户机(即我们今天所知道的客户机)开始腾飞。

在90年代中期，ICQ(1996)是第一个，其次是AOL Instant Messenger (AOL Instant Messenger, 1997)。美国在线后来收购了ICQ的创始人米拉比利斯。几年后，ICQ(现为AOL所有)获得了美国政府颁发的两项即时通讯专利专利办公室。与此同时，其他公司开发了自己的应用程序(Excite、MSN、Ubique和Yahoo)，每个应用程序都有自己的专有协议和客户端;因此，如果用户希望使用多个客户机应用程序，就必须运行多个客户机应用程序。1998年，IBM发布了IBM Lotus Sametime，该产品基于IBM收购基于haifa的Ubique和基于lexington的Databeam时获得的技术。

2000年，一个名为Jabber的开源应用程序和基于开放标准的协议启动。Jabber服务器可以充当到其他IM协议的网关，从而减少运行多个客户机的需要。通过为每个协议使用额外的本地库，Digsby、Pidgin、Trillian、Adium和Miranda等多协议客户机可以使用任何流行的IM协议。IBM Lotus Sametime 2007年11月的版本增加了对XMPP的IBM Lotus Sametime网关支持。

最近，许多即时通讯服务已开始提供视讯会议功能、IP电话(VoIP)和网络会议服务。Web会议服务集成了视频会议和即时消息传递功能。一些较新的即时通讯公司正在为语音和视频功能提供桌面共享、IP广播和IPTV。

“即时通讯”一词是时代华纳[3]的一种服务标志，在美国不属于AOL的软件不得使用。因此，即时通讯客户端Gaim(前身为Gaim或Gaim)在2007年4月宣布将更名为“Pidgin”

合作

标准的免费即时消息应用程序提供了文件传输、联系人列表、同步对话等功能。这些可能是小型企业需要的所有功能，但大型组织将需要能够协同工作的更复杂的应用程序。找到具有此功能的应用程序的解决方案是使用即时消息传递应用程序的企业版本。这些名称包括Jabber、Lotus Sametime、Microsoft Office Communicator等，它们通常与工作流系统等其他企业应用程序集成。这些企业应用程序或企业应用程序集成(enterprise Application Integration, EAI)是根据特定的约束构建的，即以公共格式存储数据。

已经有几次试图创建一个统一的标准,即时消息:IETF的SIP(会话初始化协议)和简单的即时通讯和存在利用扩展(SIP),先端(应用程序交换),拘谨的(存在和即时消息传递协议),开放的基于xml的XMPP(可扩展消息传递和到场协议),俗称Jabber和OMA(开放移动联盟)小鬼(即时通讯和业务服务)专为移动设备创建。

大多数尝试为主要的IM提供商(AOL, Yahoo!微软(Microsoft)则失败了，两家公司都继续使用自己的专有协议。

然而，当IETF的讨论陷入僵局时，路透社的协作服务主管David Gurle(微软实时通信和协作业务的创始人)在2003年9月签署了第一个跨服务提供商的连接协议。这项协议使得AIM、ICQ和MSN Messenger的用户可以与路透社(Reuters)消息服务的对等方进行交流，反之亦然。接下来是微软、雅虎!AOL与微软达成了一项协议，微软的Live Communication Server 2005用户也有可能与公众即时通讯用户对话。该协议将SIP/SIMPLE确立为协议互操作性和为访问公共即时消息云建立连接费。另外，2005年10月13日微软和雅虎!宣布到2006年(北半球)夏季，他们将使用SIP/SIMPLE进行互操作，随后在2005年12月，美国在线和谷歌达成战略合作协议，谷歌通话用户将能够与AIM和ICQ用户通话，前提是他们在美国在线拥有身份。

有两种方法可以组合许多不同的协议:

一种方法是在IM客户机应用程序中组合许多不同的协议。

另一种方法是在IM服务器应用程序中组合许多不同的协议。这种方法将与其他服务的通信任务转移到服务器。客户端不需要知道或关心其他IM协议。例如，LCS 2005公共IM连接。这种方法在Jabber/XMPP服务器中很流行，但是所谓的传输项目与任何涉及封闭协议或格式的其他项目面临相同的逆向工程困难。

Sonork所采用的一些方法,比如,企业即时通讯软件或Jabber / XMPP网络或Winpopup局域网信使,允许组织创建自己的私人即时通讯网络,使得他们能够限制访问服务器(通常与IM网络完全落后于他们的防火墙)和管理用户权限。其他公司的消息传递系统也允许注册用户通过使用基于HTTPS的安全防火墙友好协议从公司局域网之外进行连接。通常，专用的企业IM服务器有几个优点，比如预先填充的联系人列表、集成的身份验证以及更好的安全性和私密性。

一些网络已经做了一些改变，以防止它们被这样的多网络IM客户端利用。例如，Trillian不得不发布几个修订和补丁，以允许其用户访问MSN、AOL和Yahoo!网络，在这些网络被改变之后。主要的IM提供商通常会引用正式协议的需要以及安全性方面的考虑作为进行这些更改的原因。

移动即时通讯

移动即时消息传递(MIM)是一种支持状态的消息传递服务，旨在将桌面消息传递体验转换为移动的使用场景。虽然几个核心理念的桌面体验一方面适用于连接移动设备,其他人不:用户通常只看手机的屏幕——存在状态变化可能发生在不同情况下发生在桌面,和一些功能限制存在基于绝大多数的移动通信设备由用户选择适合自己的手掌。一些相关的形式和流动性差异需要考虑为了创建非常充足,强大而方便的移动体验:广播带宽、内存大小、可用性的媒体格式,基于键盘的输入,屏幕输出,CPU性能和电池供电是核心问题,桌面设备用户甚至游牧用户提供网络连接。

朋友之间网络

即时消息传递可以在朋友对朋友的网络中完成，其中每个节点连接到friendslist上的朋友。这允许与朋友的朋友进行通信，并建立聊天室，以便与该网络上的所有朋友进行即时消息交流。

情绪通常用速记来表达。例如;哈哈但是现在有一项运动正在进行中，以更准确地表达情感。实时反应，如(咯咯)(哼)(大笑)(翻白眼)正在迅速取代缩略词。

业务应用程序

事实证明，即时消息传递类似于个人电脑、电子邮件和万维网，其作为业务通信媒介的采用主要是由在工作中使用消费者软件的员工个人推动的，而不是由企业信息技术部门的正式授权或提供。数以千万计的正在使用的消费者即时通讯帐户正被公司和其他组织的雇员用于商业目的。

为了满足对业务级IM的需求以及确保安全性和法律遵从性的需要，Lotus Software在1998年发布IBM Lotus Sametime时创建了一种新的即时消息传递类型，称为“企业即时消息传递”(“EIM”)。此后不久，微软也紧随其后推出了Microsoft Exchange即时消息传递，随后创建了一个名为Microsoft Office Live Communications Server的新平台，并于2007年10月发布了Office Communications Server 2007。IBM Lotus和Microsoft都在其EIM系统和一些公共IM网络之间引入了联合，这样员工就可以对其内部EIM系统和AOL、MSN和Yahoo上的联系人使用单一接口。目前领先的EIM平台包括IBM Lotus Sametime、Microsoft Office通信服务器和Jabber XCP。此外，专注于行业的EIM平台，如Pivot Incorporated的IMtrader、Reuters Messaging和Bloomberg Messaging，为金融服务公司提供了增强的IM功能。

在IT组织控制之外的企业网络中采用IM会给那些不能有效管理和支持IM使用的公司带来风险和责任。公司实现专门的IM归档和安全产品和服务，如来自Secure Computing、Akonix、SurfControl和ScanSafe的产品和服务，以减轻这些风险，并为员工提供安全、安全、高效的即时消息传递功能。

企业实际应用

流行的IM技术用于共享信息，这很快导致组织采用IM解决方案，以获得it可能带来的优势。随着组织变得越来越基于信息(McNurlin & Sprague, 2006，第499页)，员工之间有效的知识共享、团队工作和协作环境的需求变得至关重要，尤其是在地理上更加分散的团队中。

典型的IM会话往往有一个特定的“字符”，它们通常很短，只涉及一个主题。媒体切换和多任务处理在整个过程中都是很常见的，然而IM也可能在已建立的同事和朋友之间使用更长时间、更断断续续的对话。Nardi等人(2000)在其关于工作场所IM使用的报告中指出了其他报告中经常提到的IM的四个主要功能，这些主要功能是:

快速提问和澄清

协调和调度任务

协调临时社交会议

与朋友和家人保持联系

IM可能最适合“快速提问和澄清”，因为这是其他报告中最常提到的属性。用户可以“快速响应，无需电话或FTF交互的开销”。例如，IDC报告说，“用户将IM视为一种快速、半永久‘闪’的媒介，这种‘闪’需要近乎即时的响应”(Isaacs et al.， 2002)。Nardi的第二次和第三次观察之所以能够实现，部分原因是IM客户机的“存在感”特性，用户知道谁“可用”。这是与有相同想法的同事最相关的物理空间作为彼此，甚至为其他媒介承担通信任务铺平了道路，如F2F或电话。这意味着任何形式的交流都可以通过IM的“在场意识”功能以某种方式得到鼓励。(Issacs et al, 2002)支持这一观点，“IM in business可能不是通信的主要工具，它可能只是另一种媒体的集合点，例如电话会议。”

Nardi的第三和第四次观察集中在IM的社交使用上，这在其他报告中也得到了广泛的宣传。IM是用来与朋友保持联系和安排社交活动的，这使得一些雇主认为它主要是用于这一目的。根据(Issacs et al, 2002)一项市场研究发现，“‘担心失去员工生产力’是企业在即时通讯方面最担心的问题”。

(Issacs et al, 2002)的研究进一步表明，这种担心是没有根据的，因为研究发现，平均“只有13%的谈话包含个人话题”，“只有6.4%的谈话完全涉及个人”。

2007年发表在《计算机媒介沟通杂志》(Journal of Computer mediation Communication)上的一项研究发现，在工作中使用即时通讯工具的员工比不使用即时通讯工具的同事受到的干扰更少。[5]的研究表明，即时通讯经常被用来代替其他更具破坏性的交流方式，如电话、电子邮件和面对面的交谈。他们发现，在使用即时通讯工具时，同事之间的对话比使用其他形式的通讯工具时更简短。由于其独特的设置，即时消息允许用户控制与同事交流的方式和时间。根据其中一名研究人员的说法，这项技术使人们能够在更方便的时间标记自己的可用性或推迟回复

IM解决方案通常可以分为两类:企业即时消息(EIM)和消费者即时消息(CIM)。企业解决方案使用内部IM服务器，但是这并不总是可行的，特别是对于预算有限的小型企业。第二种选择是使用CIM，它的优点是实现成本低廉，并且几乎不需要投资于新的硬件或服务器软件。然而，近年来出现了Jabber等开源IM客户端，它们提供免费的EIM级别解决方案。

附录C 软件使用说明书

附录D 主要源代码

public class RegisterAcitivity extends Activity {  
  
 private Button concel;  
 private Button confirm;  
 private EditText et\_register\_count;  
 private EditText et\_register\_password;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_register\_acitivity*);  
  
 initData();  
 //取消点击事件  
 concel.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
  
 finish();  
 }  
 });  
 //确定点击事件  
 confirm.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 register();  
 }  
 });  
 }  
 //注册  
 private void register() {  
 final String account = et\_register\_count.getText().toString();  
 final String password = et\_register\_password.getText().toString();  
 if (TextUtils.*isEmpty*(account) || TextUtils.*isEmpty*(password)){  
 Toast.*makeText*(this,"用户名或密码不能为空",Toast.*LENGTH\_SHORT*).show();  
 return;  
 }  
 Model.*getInstence*().getallthreadpool().execute(new Runnable() {  
 @Override  
 public void run() {  
 try {  
 //注册账号  
 EMClient.*getInstance*().createAccount(account,password);  
 runOnUiThread(new Runnable() {  
 @Override  
 public void run() {  
 Model.*getInstence*().getUserAccountTableDao().addAccount(new UserInfo(account,password));  
 Toast.*makeText*(RegisterAcitivity.this,"注册成功",Toast.*LENGTH\_SHORT*).show();  
 finish();  
 }  
 });  
 } catch (final HyphenateException e) {  
 e.printStackTrace();  
 runOnUiThread(new Runnable() {  
 @Override  
 public void run() {  
 Toast.*makeText*(RegisterAcitivity.this,"注册失败"+e.toString(),Toast.*LENGTH\_SHORT*).show();  
 Log.*d*("feng","注册失败"+e.toString());  
 }  
 });  
 }  
 }  
 });  
 }  
  
 private void initData() {  
 concel = findViewById(R.id.concel);  
 confirm = findViewById(R.id.confirm);  
 et\_register\_count = findViewById(R.id.et\_register\_count);  
 et\_register\_password = findViewById(R.id.et\_register\_password);  
 }  
}

public class LoginActivity extends Activity {  
 private EditText user\_edit;  
 private EditText password\_edit;  
 private Button bt\_login;  
 private Button bt\_register;  
 private Button bt\_return\_password;  
 private CheckBox cb\_login\_displaypassword;  
 private Context mContext;  
  
 public LoginActivity() {  
 }  
  
 public LoginActivity(Context Context) {  
 mContext = Context;  
 }  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_login*);  
 //初始化登登录页面控件  
 initView();  
 //监听登录按钮的点击事件  
 bt\_login.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 login();  
 }  
 });  
 //监听注册按钮的点击事件  
 bt\_register.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Intent intent = new Intent(LoginActivity.this,RegisterAcitivity.class);  
 startActivity(intent);  
 }  
 });  
 //监听找回密码按钮的点击事件  
 bt\_return\_password.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Intent intent = new Intent(LoginActivity.this,PassWordReturnActivity.class);  
 startActivity(intent);  
 }  
 });  
 //监听显示密码选择框的点击事件  
 cb\_login\_displaypassword.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 if (cb\_login\_displaypassword.isChecked()){  
 //密码明文显示  
 password\_edit.setTransformationMethod(HideReturnsTransformationMethod.*getInstance*());  
 }else{  
 //密码密文显示  
 password\_edit.setTransformationMethod(PasswordTransformationMethod.*getInstance*());  
 }  
 }  
 });  
  
 }  
  
 private void login() {  
 final String LoginAccount = user\_edit.getText().toString();  
 final String LoginPassword = password\_edit.getText().toString();  
  
 if (TextUtils.*isEmpty*(LoginAccount) || TextUtils.*isEmpty*(LoginPassword)){  
 Toast.*makeText*(LoginActivity.this,"用户名或密码不能为空",Toast.*LENGTH\_SHORT*).show();  
 return;  
 }  
  
 Model.*getInstence*().getallthreadpool().execute(new Runnable() {  
 @Override  
 public void run() {  
 EMClient.*getInstance*().login(LoginAccount, LoginPassword, new EMCallBack() {  
 @Override  
 public void onSuccess() {  
 Model.*getInstence*().loginSuccess(new UserInfo(LoginAccount));  
 EMClient.*getInstance*().groupManager().loadAllGroups();  
 EMClient.*getInstance*().chatManager().loadAllConversations();  
 //更新用户信息到本地数据库  
 //Model.getInstence().getUserAccountTableDao().addAccount(new UserInfo(LoginAccount));  
 runOnUiThread(new Runnable() {  
 @Override  
 public void run() {  
 Toast.*makeText*(LoginActivity.this,"登录成功",Toast.*LENGTH\_SHORT*).show();  
 Intent intent = new Intent(LoginActivity.this,MainActivity.class);  
 startActivity(intent);  
 finish();  
 }  
 });  
 }  
  
 @Override  
 public void onError(int i, final String s) {  
 runOnUiThread(new Runnable() {  
 @Override  
 public void run() {  
 Toast.*makeText*(LoginActivity.this,"登录失败"+s,Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
 }  
  
 @Override  
 public void onProgress(int i, String s) {  
  
 }  
 });  
 }  
 });  
  
 }  
  
 private void initView() {  
 user\_edit = findViewById(R.id.*user\_edit*);  
 password\_edit = findViewById(R.id.*password\_edit*);  
 password\_edit.setTransformationMethod(PasswordTransformationMethod.*getInstance*());  
 bt\_login = findViewById(R.id.*bt\_login*);  
 bt\_register = findViewById(R.id.*bt\_register*);  
 bt\_return\_password = findViewById(R.id.*bt\_return\_password*);  
 cb\_login\_displaypassword = findViewById(R.id.*cb\_login\_displaypassword*);  
  
 }  
}

public class InvitationTableDao {  
  
 private DBhelper mhelper;  
 public InvitationTableDao(DBhelper dBhelper) {  
 mhelper = dBhelper;  
 }  
  
 public void addInvitation(InvitationInfo invitationInfo){  
 SQLiteDatabase db = mhelper.getReadableDatabase();  
 ContentValues valus = new ContentValues();  
 valus.put(InvitationTable.*C0L\_REASON*,invitationInfo.getInvitationReason());  
 valus.put(InvitationTable.*COL\_STATUS*,invitationInfo.getInvitationStatus().ordinal());  
  
 UserInfo userInfo = invitationInfo.getUserInfo();  
 if (userInfo != null){  
 //联系人  
 valus.put(InvitationTable.*COL\_USER\_ID*,invitationInfo.getUserInfo().getUserId());  
 valus.put(InvitationTable.*COL\_USER\_NAME*,invitationInfo.getUserInfo().getName());  
 }else{  
 //群主  
 valus.put(InvitationTable.*COL\_GROUP\_ID*,invitationInfo.getGroupInfo().getGroupId());  
 valus.put(InvitationTable.*COL\_GROUP\_NAME*,invitationInfo.getGroupInfo().getGroupName());  
 valus.put(InvitationTable.*COL\_USER\_ID*,invitationInfo.getGroupInfo().getGroup\_invitePerson());  
 }  
 db.replace(InvitationTable.*TABLE\_NAME*,null,valus);  
 }  
  
 public List<InvitationInfo> getInvitation(){  
 SQLiteDatabase db = mhelper.getReadableDatabase();  
 String sql = " select \* from " + InvitationTable.*TABLE\_NAME*;  
 Cursor cursor = db.rawQuery(sql, null);  
  
 List<InvitationInfo> invitationInfos = new ArrayList<>();  
 while (cursor.moveToNext()){  
 InvitationInfo invitationInfo = new InvitationInfo();  
  
 invitationInfo.setInvitationReason(cursor.getString(cursor.getColumnIndex(InvitationTable.*C0L\_REASON*)));  
 invitationInfo.setInvitationStatus(initInvitationStatus(cursor.getInt(cursor.getColumnIndex(InvitationTable.*COL\_STATUS*))));  
  
 String groupid = cursor.getString(cursor.getColumnIndex(InvitationTable.*COL\_GROUP\_ID*));  
 if (groupid != null){  
 //群组  
 GroupInfo groupInfo = new GroupInfo();  
 groupInfo.setGroupId(cursor.getString(cursor.getColumnIndex(InvitationTable.*COL\_GROUP\_ID*)));  
 groupInfo.setGroupName(cursor.getString(cursor.getColumnIndex(InvitationTable.*COL\_GROUP\_NAME*)));  
 groupInfo.setGroup\_invitePerson(cursor.getString(cursor.getColumnIndex(InvitationTable.*COL\_USER\_ID*)));  
 invitationInfo.setGroupInfo(groupInfo);  
 }else{  
 //联系人  
 UserInfo userInfo = new UserInfo();  
 userInfo.setUserId(cursor.getString(cursor.getColumnIndex(InvitationTable.*COL\_USER\_ID*)));  
 userInfo.setName(cursor.getString(cursor.getColumnIndex(InvitationTable.*COL\_USER\_NAME*)));  
 userInfo.setNick(cursor.getString(cursor.getColumnIndex(InvitationTable.*COL\_USER\_NAME*)));  
 invitationInfo.setUserInfo(userInfo);  
 }  
 invitationInfos.add(invitationInfo);  
 }  
 cursor.close();  
 return invitationInfos;  
 }  
  
 public InvitationInfo.InvitationStatusn initInvitationStatus(int status){  
 if (status == InvitationInfo.InvitationStatusn.*NEW\_INVITATION*.ordinal()){  
 return InvitationInfo.InvitationStatusn.*NEW\_INVITATION*;  
 }  
 if (status == InvitationInfo.InvitationStatusn.*INVITATION\_ACCEPT*.ordinal()){  
 return InvitationInfo.InvitationStatusn.*INVITATION\_ACCEPT*;  
 }  
 if (status == InvitationInfo.InvitationStatusn.*ACCEPTE\_GROUP\_APPLICATION*.ordinal()){  
 return InvitationInfo.InvitationStatusn.*ACCEPTE\_GROUP\_APPLICATION*;  
 }  
 if (status == InvitationInfo.InvitationStatusn.*ACCEPTE\_GROUP\_INVITATION*.ordinal()){  
 return InvitationInfo.InvitationStatusn.*ACCEPTE\_GROUP\_INVITATION*;  
 }  
 if (status == InvitationInfo.InvitationStatusn.*GROUP\_APPLICATION\_ACCEPTED*.ordinal()){  
 return InvitationInfo.InvitationStatusn.*GROUP\_APPLICATION\_ACCEPTED*;  
 }  
 if (status == InvitationInfo.InvitationStatusn.*GROUP\_APPLICATION\_REJECTED*.ordinal()){  
 return InvitationInfo.InvitationStatusn.*GROUP\_APPLICATION\_REJECTED*;  
 }  
 if (status == InvitationInfo.InvitationStatusn.*INVITATION\_ACCEPTED\_BY\_PEER*.ordinal()){  
 return InvitationInfo.InvitationStatusn.*INVITATION\_ACCEPTED\_BY\_PEER*;  
 }  
 if (status == InvitationInfo.InvitationStatusn.*NEW\_GROUP\_APPLICATION*.ordinal()){  
 return InvitationInfo.InvitationStatusn.*NEW\_GROUP\_APPLICATION*;  
 }  
 if (status == InvitationInfo.InvitationStatusn.*NEW\_GROUP\_INVITATION*.ordinal()){  
 return InvitationInfo.InvitationStatusn.*NEW\_GROUP\_INVITATION*;  
 }  
 if (status == InvitationInfo.InvitationStatusn.*REJECT\_GROUP\_APPLICATION*.ordinal()){  
 return InvitationInfo.InvitationStatusn.*REJECT\_GROUP\_APPLICATION*;  
 }  
 if (status == InvitationInfo.InvitationStatusn.*REJECT\_GROUP\_INVITATION*.ordinal()){  
 return InvitationInfo.InvitationStatusn.*REJECT\_GROUP\_INVITATION*;  
 }  
 if (status == InvitationInfo.InvitationStatusn.*GROUP\_INVITATION\_REJECT*.ordinal()){  
 return InvitationInfo.InvitationStatusn.*GROUP\_INVITATION\_REJECT*;  
 }  
 if (status == InvitationInfo.InvitationStatusn.*GROUP\_INVITATION\_ACCEPTED*.ordinal()){  
 return InvitationInfo.InvitationStatusn.*GROUP\_INVITATION\_ACCEPTED*;  
 }  
 return null;  
 }  
  
 public void removeInvitation(String id){  
 if (id == null)  
 return;  
 SQLiteDatabase db = mhelper.getReadableDatabase();  
  
 db.delete(InvitationTable.*TABLE\_NAME*,InvitationTable.*COL\_USER\_ID* + " =?",new String[]{id});  
 }  
  
 public void updateInvitationStatus(InvitationInfo.InvitationStatusn invitationStatusn,String id){  
 if (id == null)  
 return;  
 SQLiteDatabase db = mhelper.getReadableDatabase();  
  
 ContentValues value = new ContentValues();  
 value.put(InvitationTable.*COL\_STATUS*,invitationStatusn.ordinal());  
 db.update(InvitationTable.*TABLE\_NAME*,value,InvitationTable.*COL\_USER\_ID* + " =?",new String[]{id});  
 }  
}