西南交通大学

硕士学位论文

基于XMPP协议企业级IM的研究与实现

姓名：付莎

申请学位级别：硕士

专业：计算机应用技术

指导教师：楼新远

20090501

摘 要

近年来即时通信技术的飞速发展使即时通信工具的应用更为广泛，给个人的网络生活、企业的日常办公都带来了极大的便利性与高效性。XMPP（eXtensible Messaging and Presence Protocol）可扩展消息与出席协议技术便是其中较为活跃的一种即时通信技术。由于即时通信工具在企业中的应用给企业的运营管理带来诸多便利，因而在企业中的应用越来越广泛，具有很高的研究与应用价值。

目前常用的即时通信软件的协议种类繁多，本文在研究比较了当前流行的几种协议之后，选用了基于可扩展标记语言 XML 的 XMPP 协议，由于其开放性、扩展性、安全性良好等诸多优势，并可以实现与使用其他协议的即时通信软件的互联互通，且发展前景良好，因而对于开发一款企业级即时通信系统有着十分明显的优势。本文从对 XMPP 协议的介绍与分析入手，首先简要介绍了 XMPP 协议及其发展，XMPP 协议的特点，然后又深入介绍了 XMPP 协议的内容：XMPP 协议的系统构架、地址空间、数据的传输结构、以及通信链路的建立过程等。在对协议进行了深入研究的基础上，根据本文的研究目标，针对企业级即时通信系统的特点进行需求分析，并设计与实现。在实现了即时通信的消息收发、名册管理、出席信息的交换等基本功能的基础上，着重研究了用户的管理控制问题、权限划分、可追溯性管理及功能性、扩展性的要求，实现了会议功能、文件传输，以及广播功能，模拟了与非 XMPP 系统进行交互的过程。在开源软件系统 Openfire 及 Gloox 库的支持基础上，最终设计并实现了一套完善的面向企业级的即时通信系统。

最后对系统进行了测试，完成了测试平台的搭建工作，建立相应的测试用例。系统实验测试的结果表明：系统功能完善、稳定，界面友好简洁，满足企业级即时通信系统的需求。

关键词：XMPP；企业级即时通信；XML；Openfire；Gloox

Abstract

In recent years, the rapid development of instant message technology has led to the extensive use of instant messaging system and brought about convenience and efficiency for the personal on-line life and daily business activities. XMPP (eXtensible Messaging and Presence Protocol) technology is just one of them. The application of instant messaging system has brought so many advantages to the operating and managing of enterprises that it has high research and application value.

At present, the commonly used instant messaging protocol is very various. After researching the popular protocols, and chooses XMPP based on extensible markup language XML, because of its openness, expansibility and security. It also can communicate with other instant messaging tools as well. So it has lots of advantages in developing new enterprise instant messaging system. This thesis begins with the introduction and analysis of XMPP protocol and then illustrates its development and characteristics. It emphasizes the content of XMPP protocol which includes XMPP generalized architecture, addressing scheme, data transfer structure, and the process of establishing communication link, etc. After the in-depth research of XMPP, the thesis analyses the needs of the enterprise instant messaging system and designs the corresponding system. Based on the fulfilled basic functions of the system such as sending and receiving message, roster management, presence information exchange, then the thesis focuses on the management of user control, division of authority, the management of traceability. And the system implements functionality and expansibility, such as conference function, file transfer, and broadcast function, and simulates the XMPP system transfers data with the non-XMPP interactive process. With the support of Openfire and Gloox, finally designed and implemented a comprehensive enterprise instant messaging system.

At last, the system test builds the test platform, establishes the test cases. The

result of the test shows that the functions are stable and user-friendly, can meet the basic demands of the enterprise instant messaging system.

Keywords: XMPP; EIM; XML; Openfire; Gloox

西南交通大学

学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权西南交通大学可以将本论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复印手段保存和汇编本学位论文。

本学位论文属于

1．保密□，在　　年解密后适用本授权书；

2．不保密☑，使用本授权书。

（请在以上方框内打“√”）

学位论文作者签名：付英　　指导老师签名：

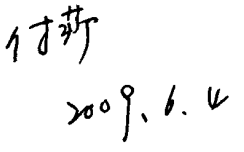
日期：2009.6.4　　日期　2009.6.4

西南交通大学学位论文创新性声明

本人郑重声明：所呈交的学位论文，是在导师指导下独立进行研究工作所得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的研究成果。对本文的研究做出贡献的个人和集体，均已在文中作了明确的说明。本人完全意识到本声明的法律结果由本人承担。

本学位论文的主要创新点如下：

1. 研究 XMPP 协议基础上，实现了基于企业级系统需求的 IM 系统，满足其管理性、功能性、扩展性等需求，并具有较高的系统安全性，并实现与其他即时通信系统的互联互通。
2. 模拟了 XMPP 系统与非 XMPP 系统之间相互交互的过程，展示了基于 XMPP 协议的 IM 系统扩展性良好，实现方便。



第1章 绪论

1.1 研究背景

随着互联网的飞速发展，Internet上的用户数成几何级数增长，越来越多的人加入到以网络为媒介的网际交流活动中，其中即时通信（Instant Messaging）工具尤其受到网络用户的青睐，它能够实时的传输文本、图片、文件，以及语音、视频等，因而成为网络用户使用频度最高的软件之一。

早期在互联网上，人们通过E-Mail、BBS等方式进行网际交流，往往不能及时把消息传递给对方，也不知道对方什么时候能够看到，或有谁能看到谁又会回应。而即时通信软件的出现解决了这一难题，它具有很高的互动性，利用它可以实时传送文字、语音信息、发送文件，以及视频通信，被公认为是现代交流方式的象征。现在，部分公司企业也开始在企业的管理中尝试加入IM的应用，在企业内部供员工之间进行交流，或异地交流联系，召开网络会议，比起传统的会议形式来说，不仅快捷，而且有效的大幅度的降低企业的运营成本。

自世界上第一款即时通信工具ICQ于1996年11月推出以来，即时通信工具由于其方便、快捷、实用、使用简单等特点，迅速得到推广，在随后短短的十几年的时间里，取得了令人瞩目的发展。迄今为止，全球约有6.02亿人使用即时通信软件进行交流。中国网民惯用的即时通信软件——“腾讯QQ”从1999年2月诞生到现在，用户规模已达到1.68亿，在我国即时通信市场上占领绝对优势[1]。

网络即时通信服务已逐步成为网络经济新的增长点，并在网络经济时代的各个新兴行业中不断壮大。据艾瑞市场咨询（iResearch）的研究报告指出2006年全球即时通信软件的用户数量已经达到4.32亿，到2010年预计全球将有6.5亿即时通信用户。艾瑞市场咨询根据eMarketer的资料整理显示，全球企业即时通信市场规模2005年为2.67亿美元，预计2010年市场将实现两倍的增长，达到6.88亿美元。到2011年，即时通信工具将成为工作人群的主要沟通方式。预计到2013年，领先跨国公司95%的职员将把即时通信软件作为他们

实时沟通交流的主要工具[1][2]。

目前，各大软件厂商都已纷纷加入IM的竞争行列，微软、IMB、谷歌、百度、腾讯等公司都先后推出了自己的IM产品，IM产品出现了多元化的竞争状态。对于绝大部分网络用户来说，IM已经不再陌生。提及IM，人们立刻会想起QQ、MSN、GTalk等即时通信工具，可以说即时通信工具已经渗透到了众多网络用户的日常生活工作中，成为了必不可少的网络工具。

企业也慢慢意识到了即时通信工具的好处，进而在企业管理中也开始使用即时通信工具，但企业用户也开始看到了针对个人用户的即时通信工具投入企业中使用所带来的弊端：一方面，即时通信工具的加入，方便了企业内部员工的日常交流，提高了员工间的协同工作的效率，降低了传统的沟通交流成本；另一方面，管理者也开始担心员工利用即时通信工具做一些与工作无关的事情降低工作效率。另外，针对个人用户的即时通信工具在其安全性、可管理性等特征上有着先天的不足。随着即时通信工具的越来越普及，针对个人用户的即时通信工具运用到企业的管理中的所带来的各种弊端也越来越明显。在这种最终用户需求变化的牵引之下，即时通信工具的用户对象开始分化，基于企业管理者的这些担心，一个新的即时通信软件类别——企业级即时通信软件（Enterprise Instant Message EIM）应运而生，并逐渐引起了企业管理者们的关注[3][4]。

企业级即时通信系统的出现，在传统的即时通信的灵活、快捷、方便等优势的基础上，更重要的在安全性、可管理性、可追溯性、可扩展性等方面都得到了一定程度的保障。由于各大厂商都有自己的IM产品，出于各自利益的考虑，大部分都采用了自己私有的通信协议，这在一定程度上阻碍了使用不同即时通信系统的企业与企业之间的沟通交流，因此在企业级即时通信的产品中急需一种使用开放式的即时通信协议的EIM以实现不同工具之间的互通互联，使EIM更加便利[5]。

美国著名未来学家约翰·奈斯比特有句名言："未来的竞争将是管理的竞争，竞争的焦点在于每个社会组织内部成员之间及其与外部组织的有效沟通上"[3]。显然，企业内外的沟通状况无疑影响着企业的管理效果，从而也决定了企业的核心竞争力。企业级的即时通信工具正是迎合了这种未来企业的管理需求，可以看出未来企业级即时通信工具必将成为新一轮互联网上即时通信工具的战场。

1.2 国内外发展现状

近几年，即时通信软件得到了突飞猛进的发展。即时通信软件系统所拥有的实时性、跨平台性、成本低、效率高等诸多优势，使之成为互联网用户最喜爱的网络沟通方式之一。在国外 MSN，AOL，Yahoo，以及最新的 Gtalk 等即时通信软件都拥有庞大的用户群。在国内，经过近十年的发展，中国互联网用户已经习惯于在网上使用即时通信软件进行交流沟通，甚至已经超过电子邮件的使用率。腾讯 QQ、MSN、阿里旺旺、飞信等国内的即时通信软件占据着绝对优势的市场分额。可以说面向个人用户的即时通信软件已经具有了广大的应用基础，且技术也已相对的成熟完善[6]。

在企业级即时通信技术领域，随着即时通信工具的逐步普及，各大软件厂商也意识到了企业级即时通信技术的市场价值及前景。尽管相对于个人即时通信工具，企业级即时通信技术才刚刚起步，但目前包括 IBM、微软等软件巨头以及国内的腾讯公司等都十分看好这个市场，各自先后推出了企业级的即时通信工具，以抢占这个新兴的市场。IBM 包含了 Lotus Sametime 的企业协作平台 IBM Lotus Notes/Domino、微软的 LCS、雅虎的 Yahoo Business Messenger、AOL 公司的 Enterprise AIM，以及国内腾讯公司的 RTX 等企业级即时通信工具相继问世。可以看出，全球各大厂商都已经意识到企业级即时通信工具的市场价值，纷纷加快脚步进入企业级即时通信市场的争夺战。

目前市场上主流的企业级即时通信解决方案主要有腾讯公司的 RTX、IBM 的 Lotus Sametime、微软的 LCS 等比较成熟的解决方案。

- 腾讯 RTX（Real Time eXchange）

RTX 是腾讯公司推出的企业级即时通信平台。企业员工可以轻松的通过服务器所配置的组织架构查找需要进行通信的人员，并采用丰富的沟通方式进行实时交流。RTX 采用的是腾讯公司的私有协议，它最大的特点在于可以与当前腾讯公司主流个人即时通信软件 QQ 进行无缝的沟通联系，且使用习惯上与 QQ 相似。由于 QQ 用户群体在国内占有绝对的优势，使腾讯 RTX 企业级即时通信软件刚一推出就占有了一定的优势，造就了腾讯 RTX 在国内的企业级即时通信软件领域占有了很大的市场分额[7]。

- IBM Lotus Sametime

IBM Lotus Sametime 是由美国 IBM 公司 Lotus 协作办公软件系统中的一

个独立模块。软件提供了整合企业即时消息传递、VoIP、视频聊天和 Web 会议等功能，并且具备业务应用所需的安全特性，在解决企业级的即时通信技术方面有一套完善的解决方案，并可以和 Lotus 内部的其它软件进行非常好的衔接配合，进行更大规模的协同办公，是一套非常值得企业管理者考虑的企业级即时通信系统。不过由于 Lotus 是一套集成度非常高的协作办公软件系统，部署成本过于昂贵，一般只有少数的大型企业能够承受其前期成本，在中小企业中应用具有一定的局限性[8]。

- LCS（Live Communications Server）

LCS（Live Communications Server）是微软公司推出的一款企业级的即时通信系统的服务器，它为企业提供了一种可管理并可扩展的企业级即时通信的解决方案。LCS 采用行业标准的 SIP/SIMPLE 协议作为通信协议的基础，必须构建于 Win2000 以上的微软平台上，且与其它微软产品例如 Office 等能达到很高的集成度。不过由于其只能运行于 Window 平台，且与 IBM Lotus Sametime 的情况相似，部署费用过高，在中小企业应用中不占有优势[9]。

除此之外，目前市场上还有一些中小型公司的产品，例如国内金 WPS Office 的金山即时通，北京点击科技的 GK－Express 竞开通信之星也都是一些面向企业级的即时通信系统，相比较之前介绍的产品，这类产品相对来说功能不如之前介绍的几款产品丰富，但部署等成本较低，在中小企业中还是具有一定优势的。

在实际应用中，企业将即时通信系统和当前企业的应用系统相互整合，然后依靠 EIM 建立统一的应用平台，有利于将公司的相关应用平台发挥出最大的应用价值。IBM 的 Lotus Sametime、微软的 LCS 都对该企业各自的办公系统以及应用软件 Lotus，以及 Office 和 MSN 进行了整合，使企业的运用效率能达到最大化，这是 EIM 发展的一个重要特征，可以和该企业的其它应用系统进行融合，也是 EIM 发展的一个趋势。

本课题所涉及的基于 XMPP 协议的即时通信系统目前属于比较新兴的技术，在个人即时通信市场上已经出现了不少产品，例如 Google 公司的 GTalk，Jive Software 公司的 Spark 等产品。随着 Google 等国际大公司的加入，基于 XMPP 协议的即时通信系统正在逐步的壮大自己的队伍，也促进了 XMPP 协议的快速发展。相对于个人即时通信的发展，基于 XMPP 协议的企业级即时通信产品相对来说很少，发展滞后，具有很大的发展空间。本课题的研究正

是出于 XMPP 协议的安全性、开放性以及强大的扩展性，在企业级即时通信系统应用上具有很高的研究与应用价值而提出的。

1.3 研究的目标与主要内容

1.3.1 研究目标

本课题研究的目标是在分析研究 XMPP 协议的基础上，基于 XMPP 协议及其扩展协议，研究并实现一套符合企业级应用需求的即时通信系统，除能够很好的满足用户的基本需求、管理需求与功能性需求外，并达到较高的安全性，可扩展性，实现与其他即时通信系统的互联互通。

1.3.2 研究的主要内容

针对本文的研究目标，本文研究的主要内容可分为以下几个方面：

1. 安全性与可追溯性方面

传统面向个人的即时通信系统在企业内部使用过程中，除正常工作使用外，企业员工也可随意的和企业内部或者外部的人员进行沟通联络。在交流的过程中，很可能将企业的商业机密信息泄露出去，给企业造成不同程度的损失。因此企业级即时通信系统必须解决这个问题，对用户间的交流范围和交流动作进行控制，实现一种企业的痕迹化管理，简单的说就是沟通记录是可以审计和追溯的。并依靠下述的可管理性要求，提升整个系统在企业运用中的安全性。

2. 可管理性与可控性方面

企业级即时通信系统在企业内部的应用必须是可控和可管理的。员工根据岗位职能的不同交流范围需要有所控制，使用的系统提供的应用功能也要有所区别，需要划分出不同的使用权限，整个系统要在管理者的掌控范围之内。本系统将考虑用户的管理、权限的分配，使可管理与可控性方面符合要求。

3. 功能性需求方面

根据企业真实的办公环境需求，系统需要提供一些个人即时通信工具不具有或使用方式不一致的针对企业用户的附加功能，提升企业用户的办公效率以及员工间协同工作的效率，体现企业级即时通信系统的优势。

4. 扩展性方面

当前企业内部应用的各种协同办公系统越来越多，在降低运营、维护成本，减少用户的培训成本等因素之上，企业的多种应用系统间的相互整合的需求也越来越明显，本系统将实现一种本企业级即时通信系统与其他系统的交互以及整合的方法。提升整个系统在真实环境下应用和推广的优势。

1.4 本文的组织结构

本论文主要是基于 XMPP 协议，研究并实现一套适合于企业使用的企业级即时通信软件，本论文共分为 6 章，其大体的结构安排如下：

第 1 章，绪论，介绍了即时通信的研究背景、国内外的研究现状，针对当前的发展现状提出问题，并提出本文的研究目标与主要内容，即通过研究 XMPP 协议，开发出一套企业级即时通信系统。

第 2 章，首先对即时通信协议 XMPP 进行了一个简要的介绍，随后介绍了 XMPP 协议的发展历史以及 XMPP 协议与 Jabber 协议之间的关系。介绍了 XMPP 协议具有的几个特点，最后对目前市场是流行两种协议进行了一个简单的比较，对了解 XMPP 协议有一个基础的认识。

第 3 章，对 XMPP 协议进行了技术层次的描述，介绍了基于 XMPP 协议的即时通信系统的构架、基本组成实体、实体之间的关系、地址空间，XMPP 的数据传输结构，最后对 XMPP 通信链接的建立过程及步骤进行了介绍，对 XMPP 协议的基本内容有了比较深入的了解。

第 4 章，对企业级即时通信系统进行了分析，分析企业级即时通信系统的系统要求，以及几个重点功能的需求分析，并提出了问题的解决思路。

第 5 章，阐述了基于 XMPP 的企业级即时通信系统的设计实现，重点包含了几个重点功能模块的详细实现过程。最终完成一个基于 XMPP 的企业级即时通信系统。

第 6 章，对系统进行测试，首先介绍了测试的环境与范围，然后对系统实现的各个功能分别进行了测试，测试表明系统功能完善，能够满足企业级用户的需求，基本达到了预期的目标。

最后对本论文的工作进行了总结，总结分析了研究成果，并对以后的研究工作提出了展望。

第2章 XMPP协议以及Jabber

2.1 XMPP协议以及Jabber

2.1.1 XMPP协议简介

XMPP（eXtensible Messaging and Presence Protocol）可扩展消息与出席协议，是一种基于可扩展标记语言XML的开放式协议。XMPP协议设计的目的是用来进行准实时的消息和出席信息以及请求—响应服务，可用于即时通信的消息传递以及在线现场探测等运用[10]。

XMPP协议由于IETF的介入，形成了即时消息和出席信息技术的新的国际规范。在当今各种即时通信工具大行其道的年代，统一出了一种通用的即时通信协议技术标准。只要遵循该标准的即时通信工具或相关扩展应用，都能够进行无障碍的交流与联系，为各企业、组织或个人提供的即时通信工具产品的互通互联，以及和其它的应用融合提供了新的标准依据，一经推出就受到了市场的青睐。

目前，基于XMPP协议的即时通信工具已经逐步向市场推广，由于其良好的特性，得到了来自例如Google等公司的支持，也进一步推动了XMPP技术的发展。Google推出的GTalk，Jive Software公司的Spack等很受欢迎的几款即时通信工具，都利用了此技术。这些国际企业的相关产品的推广都为XMPP技术的发展提供了强有力的支持，并且目前有不少组织和个人热衷于XMPP协议的不断完善和改进，IETF的XMPP工作组也随着市场的需求相继制定出各方面的扩展协议标准，完善XMPP协议的运用。可以看出，XMPP有着良好的发展前景。

2.1.2 XMPP协议与Jabber协议

XMPP协议的前身是出现于1999年的开源社区的Jabber协议。由于这一层特殊的关系，因此现如今“Jabber”和“XMPP”协议往往在论文或参考文献交替使用。本文中提及的Jabber和XMPP协议其实也是同一种协议在不

同时期的不同名称而已。

1998 年，Jeremie Miller 发起了一个免费的开源的 Jabber 协议项目，他厌倦了当时使用四种不同协议的客户端，因此，他决定“从零开始”设计出一个真正开源的统一的即时通信协议，使不同的即时通信客户端能够互通互联，从根本上解决这种协议使用混乱的问题，由于 Jeremie Miller 的努力，Jabber 协议出现了[11]。

经过几年的发展，Jabber 协议逐渐成熟，基于 Jabber 的应用也越来越多，受到了很多方面的关注，于是 2001 年 Jabber Software Foundation(JSF)，Jabber 软件基金会正式成立，专门用于协调越来越多的基于 Jabber 的开源项目和一些商业的应用。

2002 年 Jabber 开源社区成员正式提交了 Jabber 协议草案给 IETF(Internet Engineering Task Force) ——Internet 工程任务组，希望能成为 IETF 的标准化协议。IETF 讨论了是否有成立专门的 IETF 工作组并对 Jabber 协议进行标准化的必要，当时为其取了一个新的中立的名称：可扩展消息与出席协议（XMPP)。由于其良好的特性和庞大的用户群，三个草案协议于当年被正式提交，XMPP 工作组也正式成立，被授权接手开发和改变 Jabber 协议以适应 IETF 的消息和出席信息技术。经过 XMPP 工作组的努力，XMPP V1.0 于 2004 年正式成为了 IETF 的即时通信和在线技术的 RFC2779 标准。同年 10 月发布了 XMPP 的相关 RFC 标准（核心协议 RFC3920、RFC3921、RFC3922[12]、RFC3923[13])，随后，IETF 宣布结束 XMPP 工作组的工作，然而 XMPP 扩展协议的工作仍然由 Jabber 软件基金会继续负责与进行着。这些扩展协议也就是 Jabber Extension Protocol（JEP)。2007 年 1 月 Jabber 软件基金会（JSF）正式更名为 XMPP 标准基金会（XMPP Standards Foundation）——XSF，JEP 也正式更名为 XEP[14]。

以上简要的描述了 Jabber 和 XMPP 协议之间的历史关系，和各种名称的变动过程。可以这样说 XMPP 协议是基于 Jabber 协议原型发展、进化和建立起来的。

2.2 XMPP 协议的特点

XMPP 协议是一种基于 XML（可扩展标记语言）流，实现任意两个网络终端准实时的交换结构化信息的通信协议。XMPP 提供一个通用的可扩展的

框架来交换 XML 数据，其主要是用来建立即时消息和出席信息应用以实现 IETF RFC2778[15]，RFC2779[16]的需求，是一种开放式的传输 XML 流化元素的协议。

- **开放性**

XMPP 最初的原型协议 Jabber 的初衷就是构建一套免费的、开源的即时通信协议。发展至今，被 IETF 订立为即时消息和出席信息技术标准，所订立的标准协议是免费的、开源的和容易理解的。任何企业和个人都可以在所需要的任何工程中使用 XMPP 协议。

- **可扩展性**

XMPP 协议是一种基于 XML 流元素的传输协议，由于所使用的 XML 技术本身就是一种极具扩展性的标记语言，而 XMPP 协议订立的特点就是结构化的传输 XML 元素。在只要满足 XMPP 协议传输的 XML 节元素结构之下，可以很容易的添加新的属性或包含新的子节点来扩展现有的协议功能。这种自定义的结构，只要是通过了通信双方的认可，都可以加入任何的新的元素或属性表达出新的特有的功能信息。XSF 正致力于订立出各种各样的 XMPP 协议的扩展协议 XEP 来不断的满足即时通信和出席信息的新的需求。XEP 协议的不断增加和完善，正是由于 XMPP 良好的扩展性。

- **安全性**

XMPP 协议的原型协议 Jabber 订立之初就已经充分的考虑到了复杂的网络传输条件下，被传输的节信息的安全性问题，因而采用了多种目前网络传输信息的安全措施保证节信息的安全。一个节信息的成功投递必须经过 TLS 安全传输层协议、SASL 简单验证和安全层协议多层的验证，采用多种强度算法如 BIGEST-MD5、KERBEROS_V4、PLAIN 等多种加密算法和签名证书认证的方法保证流传输的安全特性。在安全性这一方面，可以说 XMPP 在目前的即时通信协议中安全性是比较高的[17]。一个 XML 流化节的传输安全认证层如图 2-1 所示：

- **平台无关性**

XMPP 协议只是一个标准，并不包含具体的实现过程，实现的过程可以在各平台下独立完成。可以运行基于 XMPP 协议的应用程序在不同的平台下，例如服务器、个人计算机、手持设备、甚至运行于信息采集、感应设备之上。

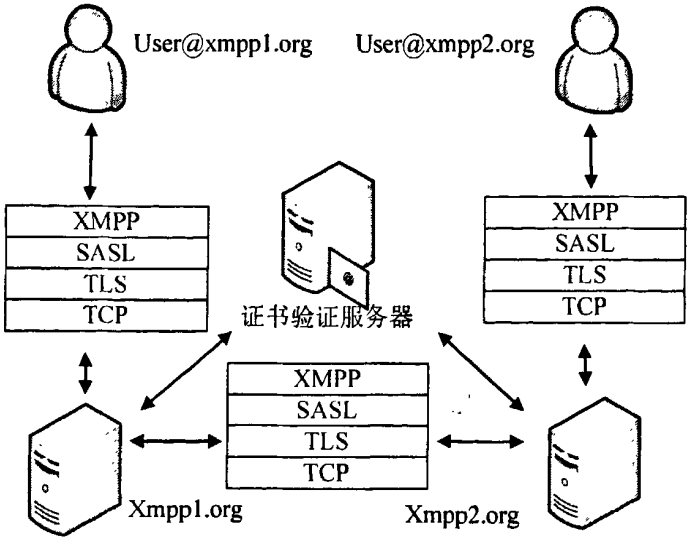


图 2-1 XML 流传输过程分层结构示意图

基于以上 XMPP 的特点，说明 XMPP 有很高的安全性、可扩展性，可以很好的满足对于开发一款企业级即时通信系统的需求，并且由于其开放性，可以起到统一即时通信协议混乱的局面，并且能够实现与其他即时通信系统的互联互通，因而选择 XMPP 协议进行企业级即时通信系统的开发，不失为一种很好的选择。

2.3 XMPP 协议与其它开放式协议的比较

目前除了 XMPP 协议外，还有一些其它开放式的协议存在，最主要的是 SIMPLE 协议[18]。

SIMPLE 协议也是 IETF 的标准协议，是一个目前为止制定的较为完善的一个即时通信协议。和 XMPP 一样，该协议也是符合 RFC2778 和 RFC2779 的要求。和 XMPP 协议基于 XML 流传输结构化数据不同的是，SIMPLE 协议是基于 SIP 协议传输即时通信数据的。SIP 最初是由 IETF 制定来为终端服务的协议，一般情况下应用在建立语音通话领域，一旦连接以后，依靠如实时协议（RTP）来进行实际上的语音发送。SIP 不仅仅能被用在语音中，也可以用于视频传输领域[19][20]。SIMPLE 是基于 SIP 协议发展起来的即时通信技术，对 SIP 进行了一定的扩展。由于 SIMPLE 协议出现的时间比 XMPP 协议出现的时间早，商用化的个人或企业即时通信系统都出现的较早，目前有微

软、IBM 等公司企业的即时通信系统，都是基于 SIMPLE 协议开发的，应用范围较广泛，技术也较成熟。

相比较而言，XMPP 技术属于新生技术，由于 XMPP 极具扩展性，在基于 XML 基础上可以很容易扩展到不同的应用和系统。在这一方面 SIMPLE 相对处于弱势。且 XMPP 以及扩展协议形成标准化协议的数量与范围更为广泛，而 SIMPLE 相应的协议可能还只是草案或者实验性的。但 XMPP 协议比较 SIMPLE 不足之处在于支持语音和视频传输方面的不足，SIMPLE 基于 SIP 协议使其在这方面具有先天优势，并且技术也比较成熟与完善，XMPP 也没有停止这方面的研究，语音和视频传输的 XMPP 扩展协议也在进一步的制定与完善之中。

由于 XMPP 在扩展性和标准化等方面的优势，得到了越来越多国际公司企业的支持，例如 Google、HP 等，商业化的脚步逐渐加快，成为了一种应用更为广泛的即时通信协议。可以说，未来 XMPP 比 SIMPLE 更具有市场优势[21]。

对比 SIMPLE 与 XMPP 协议，如表 2-1 所示：

表 2-1 SIMPLE 与 XMPP 对比

协议	SIMPLE	XMPP
基础	SIP 协议	XML 协议
功能	支持各种即时消息通信	支持各种即时消息通信
扩展能力	一般	很强
主流厂商支持	微软、IBM 等	Google、HP 等
前景	已率先广泛应用	后来居上

2.4 本章总结

本章首先对即时通信协议 XMPP 进行了一个简要的介绍，随后介绍了 XMPP 协议的发展历史，XMPP 协议与 Jabber 协议之间的关系。介绍了一下 XMPP 协议具有的几个特点，最后对目前流行的 XMPP 协议与 SIMPLE 协议进行了一个简单的比较，列举出了两个协议之间的优缺点，对了解 XMPP 协议有一个基础的认识。

第3章 基于XMPP协议即时通信系统分析

3.1 系统构架

尽管在XMPP协议的RFC文档中，没有规定XMPP必须使用特定的网络结构，但通常情况下，XMPP被公认并最终实现为一种类似于Email系统的分布式网络结构。

正常情况下，一个终端客户只与它注册成功的XMPP服务端进行通信，不与外部的其它实体进行通信。两个终端客户之间的通信方式都是由各自终端分别与各自注册的服务端通信，两个服务端再分别进行互联通信，中转两个域客户端之间传输的信息，如果为同一域的两个终端客户，服务端为同一个的情况下，终端客户传输的信息也必须发送到服务端中转，而两个终端客户不进行直接的通信。像Email系统一样，属于一种客户端—服务器的构架模型[22]。

XMPP系统主要由四部分组成[10]：

1. XMPP服务器：XMPP服务器充当XMPP通信的一个智能抽象层，主要负责管理发出的连接或者与其它实体的会话，接收或者转发XML流元素给授权的客户端、服务器或者其它实体。

2. XMPP客户终端：大部分的客户终端直接与服务器相连，通过XMPP获得由服务器或任何其它相关的服务所提供的全部功能。多个不同资源的客户端可以同时登录并且并发的连接到一个服务器，每个不同资源的客户终端通过XMPP地址的资源标识符来区分。

3. XMPP网关：网关是一种特殊用途的服务器端服务，主要功能是把XMPP协议传输的信息翻译成外部（非XMPP系统）消息系统所能识别的信息，并把返回的消息翻译成XMPP信息。例如翻译成MSN通信消息，Email消息等。

4. XMPP网络：XMPP服务器都是由一个唯一的网络地址来标识的，服务器与服务器之间的通信是一种直接的客户端到服务器端通信模式的扩展，实际上整个系统是由很多互通的服务器构成，任意两个服务器间的通信都是可选的。这种模式常见于那些需要网络地址标准化的协议（例如SMTP协议），

RFC3920中明确定义了XMPP服务器的标准化地址。使XMPP服务器间构成了一个特定的XMPP网络。

具体构架模型如图3-1所示：

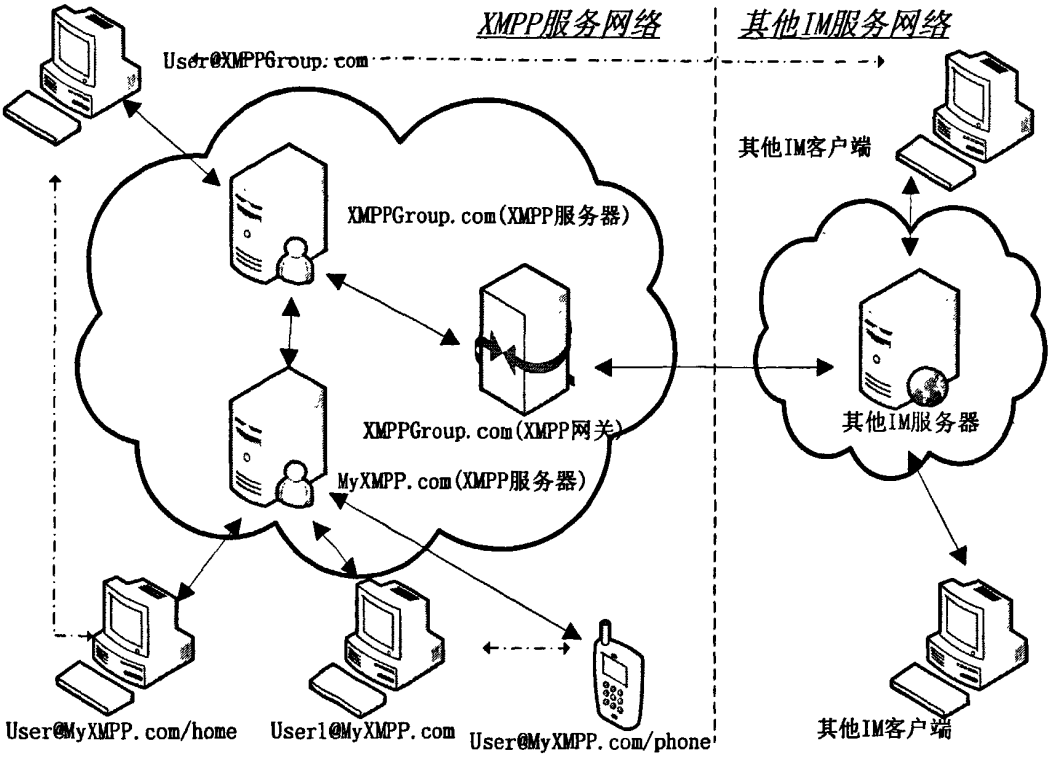


图 3-1 XMPP 网络拓扑图

如图3-1所示，是一个多网络通信的模型图。图示左边为XMPP服务网络，右边为其它的IM服务网络。在XMPP服务网络中，有两台XMPP服务器，四台XMPP服务终端，和一台XMPP网关。在其它IM服务网络中有一台其它的IM服务器，两台其它IM服务的客户端。

在XMPP服务网络中，四台XMPP服务终端，分别属于两个域XMPPGroup.Com与MyXmpp.Com，图中实线连接的两个终点表示的是真实的物理通信链路，虚线连接的两个终点表示抽象层上两个终端之间的通信链路[23]。

- XMPP网络内的终端通信

假设User@XMPPGroup.com终端客户希望与User@MyXMPP.com/home终端客户进行通信，首先User@XMPPGroup.com终端客户发送的消息必须首先发送给XMPPGroup.com这台服务器，又由这台服务器转发给User@MyXM

PP.com/home终端客户所在的MyXMPP.com服务器，最后又由MyXMPP.com服务器转发给User@MyXMPP.com/home终端客户。在物理链路上，此次通信经过了其它两个端点的中转，而不是两终端客户间的直接通信，这一点与大部分现有的即时通信系统两终端客户之间直接进行物理链路通信不同，在安全性上有一定的提升，确保了通信双方都必须通过验证才可以进行通信。

- XMPP网络与非XMPP网络的终端通信

假设终端客户User@XMPPGroup.com/home希望与其它IM系统中的终端客户进行通信，此时User@XMPPGroup.com/home终端客户第一步是将需要传送的信息发送给自己域的XMPP服务器XMPPGroup.com，服务器XMPPGroup.com查询到发送信息的接收方不在XMPP网络中，即在服务器本身的服务中查找是否有相应的网关可以将XMPP协议翻译成外部IM所能识别的信息，最终服务器XMPPGroup.com将终端客户发送的消息递交给网关XMPPGroup.com服务器处理，网关再将翻译后的信息传递到其它非XMPP网络中的终端客户一方，达到双方通信的目的。

- 实体通过不同资源的通信

假设User@XMPPGroup.com/phone是XMPP中终端用户User@XMPPGroup.com的另一个实体，通过资源名称“/phone”标识了与“/home”的不同。XMPP系统正是通过这种利用资源名称的不同，标识了同一帐户可以在同一时间与其它终端实体进行通信而互不影响。在此模型图中终端客户User@XMPPGroup.com/phone与User1@MyXMPP.com相互通信的过程中，并不影响同一用户的另外一个实体User@XMPPGroup.com/home与其它终端客户之间的通信。

虽然在XMPP规范文档中未给出XMPP的具体模型结构来涉及到两实体客户间是否可以直接通信的问题，但在现有的所有实现中除了很少一部分的运用（例如文件、音视频的传输，涉及到效率的问题，在XMPP的扩展协议中，有时可能采取的最终实现模型为两实体客户端的直接通行）外，都采用类似于Email系统的构架，所有实体客户端所需要对另一个实体客户端进行的通信信息，都是通过服务器中转完成。

3.2 地址空间

XMPP网络中，只要是通过XMPP协议通信的实体都具有一个唯一性的标

识地址，这个标识地址是由RFC2396规范所要求的必须格式。由于历史的原因，在XMPP网络中一个标识地址被称作为Jabber Identifier或JID。一个合法的唯一标识地址JID通常会由节点名、域名、资源名三部分组成，类似<user@domain/resource>的形式出现[10]。

- **域名**

域名是整个JID中唯一必须的元素，通常的情况下一个域名代表XMPP网络中的一个服务器或者网关，其它实体通过连接它来实现XML流数据的转发和数据管理功能。它也可能是一个XMPP服务器的子域名，包含了其它的额外服务（例如多用户的聊天服务，用户目录等）。域名可以是一个IP地址、IANA（国际互联网代理成员管理局）注册定义过的国际化域名，也可以是一个服务主机的主机名称。

- **节点名**

节点名是一个可选的标识符，例如上例中的“user”。通常情况下节点名和域名相组合，标识了一个需要向服务器或网关请求服务的终端客户实体。此外也有可能标识的是一个XMPP服务器一个服务的实体，例如一个房间的地址。

- **资源名**

资源名是一个可选的组成部分，放在域名后面并由符号“/”进行分割，资源名可以跟在<node@domain>后面也可以跟在<domain> 后面，它通常表示一个特定的会话、连接（比如设备或者所在位置），或者附属于某个节点ID相关实体的对象（比如多用户聊天室中的一个参加者）。对于服务器和其他客户端来说，资源名是不透明的，通常情况由终端客户端来维护资源名。一个实体可以并发维护多个已连接的资源，每个已连接的资源由不同的资源名来区分。

3.3 XMPP 数据传输结构

XMPP协议是基于XML流结构化数据传输的形式进行信息传递的。XML (Extensible Markup Language)是一种可扩展标记语言。而XMPP正是基于XML的开放式即时通信协议。在XMPP中关于XML的定义有两个：XML流和XML节[24][25]。

3.3.1 XML 流

XML流是一个容器，包含了两个实体之间通过网络交换的XML元素。一个XML流是由一个XML打开标签 <stream> (包含适当的属性和名字空间声明)开始的，XML流的结尾则是一个XML的关闭标签 </stream> 。在流的整个生命周期，初始化它的实体可以通过流发送大量的XML元素，用于流的控制信息和XML节信息。

3.3.2 XML 节

一个XML节是一个实体通过 XML 流向另一个实体发送的结构化信息中的一个离散的语义单位。一个XML节直接存在于根元素<stream/>的下一级，通常情况下XML节是一个完整均衡的元素，拥有打开和关闭标签。XMPP协议中，除了XMPP建立连接时进行的TLS询问交互，SASL用户身份验证，其它所有的信息传递都属于XML节。

在所有的XML节元素中一般都含有以下三种最常用的属性值：

（1）to属性：标识此XML信息节发送到的目的实体地址。

（2）from属性：标识此XML信息节发送者的实际地址。

（3）xmlns属性：标识此XML节具体使用的子元素或属性来自哪一个命名空间，是XML节一个重要的属性值。在XMPP标准协议中每一个XML节元素的命名空间都是标准化定义的，但是在XMPP的扩展协议中，通信双方可能支持不同的XMPP扩展协议，此时只能通过xmlns（xml name space）属性值来确认，通信双方是否能识别对方所发送的XML节所表达的意思。xmlns是每一个XML节元素都必须指明的属性值（除了以下将介绍的三种类型的XMPP已经定义的主要节元素拥有默认的命名空间之外，其它元素都必须明确的标示出xmlns属性）。

在XMPP通信中大量存在的主要的三种类型的节（Stanza）分别为<presence/>节、<message/>节以及<iq/>节。

- <presence/>

<presence/>出席信息节，用于标识一个实体或者资源终端客户的出席信息情况，该元素可以被看作一个基本的广播或“出席—订阅”机制，用于多个实体接收某个已订阅的实体的信息。通常情况下，一个实体应该不带“to”

属性发送一个出席信息，这时这个实体所连接的服务器应该广播或多播那个节给所有订阅的实体。无论如何，一个发行实体也可以带“to”属性发送一个出席信息节，这时服务器应该路由或递送这个节给预定的接收者。

示例：

```
<presence>
    <show>chat</show>
    <status>call me</status>
</presence>
```

该示例中展示了一个基本的<presence/>节的构成，其中包含子元素<show/>节，是一个非人类可读的XML字符数据表达特定的实体或资源的可用性状态。值为chat，表明了该发送者当前的状态为在线激活状态，在无权限限制或者屏蔽通信的情况下，当前可以和任何人进行沟通联系。<status/>子元素内容节则是<show/>节的一个详细描述，提供了供接收方可阅读的详细的可用状态信息。此节示例展示的含义为发送的实体或资源此时已经激活，可以和其它用户交流。

<presence/>的使用范围：<presence/>不仅仅用于上例所示的标识一个实体或者资源终端客户的出席信息情况，在订阅其它终端客户的出席信息时，或者在收到一个其它终端客户出席信息的订阅的处理（接受或拒绝）时，取消先前的出席信息订阅，以及在另一个实体端取消对自身的出席信息订阅时都是使用<presence/>进行传输和处理的。

- <message/>

<message/>消息节用于XMPP中信息的传递以及交换。它可以被看作是一个“push”机制，用于一个实体推送信息给另一个实体。该元素能准确的表达出通信双方传递的信息内容，以及传递的源和目的地址。

示例：

```
<message to="remote@xmpp.com" from="example@xmpp.com"
  type="chat" xml:lang="cn">
      <body>Hi,Remote</body>
</message>
```

该示例展示了一个基本的<message/>节的构成，属性值to、from、xml:lang以及type分别代表了该信息节发送到的目的地、源地址，所使用的语言以及

消息节的类型。其中包含了一个<body/>子元素节， 其值就是此次信息的发送内容。此节示例展示的含义为一个实体终端用户example@xmpp.com向其另一个提示终端用户remote@xmpp.com发送了一条“Hi,Remote”的消息。

<message/>的使用范围：在XMPP协议的使用中，涉及到最终可供人类阅读理解的消息的传递以及交换都是使用此节完成的。

- <iq/>

<iq/>节（Info/Query），是一个“请求—应答”机制，使一个实体能够向另一个实体请求信息并做出应答。请求和应答所包含的数据定义在IQ元素的一个直接的子元素的名字空间声明中，以区分请求或应答的不同内容。在XMPP中，请求用户的联系人名册等信息所用的就是<iq/>节。

示例：

```
<iq from="example@xmpp.com/home" type="get" id="roster_1">
        <query xmlns="jabber:iq:roster" />
<iq/>
```

该示例展示了一个基本的<iq/>节的构成，其中的type属性值代表了此<iq/>节的动作，在此为获取操作。id属性值用于一个线索跟踪（当此次请求完成时，服务端回复的信息type属性值为“result”表明</iq>节的类型是回复一个先前的请求。而id属性值与请求时的id值相同，为“roster_1”，则表明回复的是哪一个请求），其中也包括了一个子元素<query/>节。该节的xmlns属性表明了此<iq/>节所需要操作的内容。<iq/>节请求和应答的信息种类很多，每一个<iq/>节的具体含义以及内容，即是通过子元素节<query/>的xmlns属性值所表示的命名空间来区分的。此示例展示的含义为终端客户“example @xmpp.com/home”向服务端请求该用户的联系人名册。

<iq/>的使用范围：<iq/>节是三种基本类型的节中使用用途最多样化的一个节。在获取用户名的联系人名册、信息的屏蔽，以及获取用户的私有数据等等方面都是使用该节来完成的。

XMPP中，涉及到通信双方的内容以及交互过程的控制，大部分情况下，都是由以上的三种类型的XML元素节构成。以上示例分别展示了三种类型节的基本用法。三种类型的节都还包括很多其它的属性以及字节元素，本文将在以下章节的再次出现中解释其用法及含义。在XEP中，很多的扩展协议也是通过在以上的三个基本元素节中增添新的子元素节来表达新的含义。

3.4 XMPP通信链路建立过程

XMPP协议中对两实体间的通信链路建立过程做出了明确的规范，任何基于XMPP协议的实体间建立通信链接必须遵循这个规范。

在XMPP规范中任意一个实体访问一个拥有服务的服务器，必须通过TCP协议建立连接（包括服务器到服务器的连接）。服务器端的访问端口为XMPP在IANA注册过的5222端口，此端口并不是强制的，但是建议使用[24]。

在TCP连接建立以后，会有一定的安全措施开始协商，以验证通信双方身份的合法性。最先开始的安全协商为通信双方协商是否使用TLS，在XMPP规范中已经明确的说明，服务器端必须支持使用TLS，以增强其安全性。在客户端，XMPP规范没有强制客户端是否使用TLS，此时客户端可以和服务器端商议是否使用TLS。不过，当前绝大部分的服务器端为了增强安全性，都必须要求客户端使用TLS。XMPP协议规范中也明确的说明服务器端到服务器端必须使用TLS。

在通信双方TLS连接建立以后，连接接受者开始验证连接发起者的身份，XMPP规范中规定了此时使用的验证方式为XMPP特定的SASL（简单验证和安全层），SASL提供了一个通用的方法为基于连接的协议增加验证支持。使用了多种验证方式例如DIGEST-MD5、KERBEROS_V4和CRAM-MD5等，供连接双方协商使用。

在SASL验证成功完成后，通信双方就可以正式的通过发送XML数据节开始通信。

以下给出一个XML数据流传输的示例，演示了一个XMPP实体客户端通过连接到服务器后给另一实体终端客户发送消息的XML流交互传输过程。示例中Sx表示服务器，Cx表示客户端。

```
C1: <?xml version="1.0"?>
    <stream:stream to="example.com" xmlns="jabber:client"
     xmlns:stream="http://etherx.jabber.org/streams" version="1.0">
S1: <?xml version="1.0"?>
    <stream:stream from="xmpp.com" id="someid"
     xmlns="jabber:client" xmlns:stream="http://etherx.jabber.org/streams"
     version="1.0">
```

……TLS 建立，SASL 验证，客户端资源绑定省略……

```
C2: <presence xml:lang="cn">
        <show>chat</show>
    </presence>
C3: <iq from="example@xmpp.com/home" type="get" id="roster_1">
        <query xmlns="jabber:iq:roster"/>
    </iq>
S2: <iq to="example@xmpp.com/home" type="result" id="roster_1">
        <query xmlns="jabber:iq:roster">
            <item jid="remote@example.com"
             name="Remote" subscription="both">
                <group>Friends</group>
            </item >
        </query>
    </iq>
C4: <message from="example@xmpp.com/home"
     to="remote@example.com" xml:lang="cn">
            <body>Hi,Remote</body>
    </message>
S3: <message from="remote@example.com"
     to="example@xmpp.com/home" xml:lang="cn">
        <body>Hi,example</body>
    </message>
C5: </stream:stream>
S4: </stream:stream>
```

C端首先发送了一个打开流标记<stream/>，对应的S端也返回一个打开流标记<stream/>，表明了C与S开始建立连接（过程C1，S1）。其后C与S分别进行了协商建立了TLS连接与SASL验证身份（本步骤省略）。在C的身份经确认之后，C首先发送了一个自身的出席信息（过程C2）给服务端S，服务端负责发送C的出席信息给C出席信息的订阅者（本过程由于是S发送给其它用户的信息，在示例中并没有显示）。其后C向服务端S请求C的名册列表（过

程C3），S端返回C的名册列表（过程S2）。之后，C开始给其它终端“remote@example.com”发送信息（过程C4），S端返回由“remote@example.com”返回的消息。最后通信结束C端与S端分别发送一个<stream/>关闭标签，关闭会话，整个交互过程结束。

通过示例可以看出在一次成功的会话中，首先通信双方必须发送一个流打开标记，表示会话的开始，之后双方协商TLS的使用，在TLS的使用协商完毕后，开始SASL身份验证过程，验证完毕后，终端客户端首先发送自身的出席信息给服务器，表明终端客户已经准备好和其它终端客户通信（此步骤不是必须的，但常规下终端客户端都会发送），之后请求终端客户的联系人名册（此步骤也不是必须的，但常规下终端客户也都会请求联系人名册，如果不请求联系人名册，终端客户无法确定可以发送信息的用户）。在获得了联系人名册后，终端客户可以选择向其中的一个用户发送消息。在成功的发送一个消息后，一次会话成功。终端客户可以选择关闭会话，也可以选择继续发送其它的信息。

3.5 本章总结

本章首先对基于XMPP协议的即时通信系统的构架进行了一个概要的分析，说明了XMPP即时通信系统的几个基本组成实体，实体之间的关系。接着对XMPP协议中涉及的地址空间进行了介绍，介绍了XMPP地址空间中的每一种地址所代表的含义以及表明的服务类型。然后对XMPP协议中传输的XML流以及XML节的数据结构进行了简单的概述，并介绍了XMPP协议中最为重要的常用的三种类型的元素节的数据结构，基本属性所表达的含义，以及其使用范围。最后对XMPP通信链接的建立过程及步骤进行了简要的介绍，并给出了一个示例展示了一次成功的会话过程中，从链路的建立、相关信息的发送以及请求，到最终的会话关闭过程中XMPP协议中涉及到的XML流处理的XML数据元素的交互过程。

第4章 企业级即时通信系统分析

4.1 企业级即时通信系统概述

当即时通信系统的应用越来越普及到企业中来时，针对个人用户的即时通信工具运用到企业中的各种弊端逐渐显露出来。企业内部的信息的安全性以及隐秘性得不到保证，随时会有泄露的危险；内部员工使用个人即时通信工具用于和其他人的沟通得不到管理者的有效管理以及控制；与企业内部其它应用软件不能相互整合，造成了培训成本、使用成本以及维护成本的增加；以及企业用户所需求的一些功能性要求在个人即时通信系统中不具备等等，针对个人用户的即时通信产品运用于企业中的弊端越来越明显，这也推动促进了企业即时通信工具发展[26]。

本文中的企业级即时通信系统与个人用户的即时通信系统的不同之处在于：对于个人即时通信系统运用到企业中所产生的各种弊端，采取相应的处理措施，解决、完善前面所阐述的各种问题。关键需要解决的几个问题是：

1. 企业员工用户的管理；
2. 企业员工用户的权限划分；
3. 企业应用即时通信系统的安全性控制；
4. 企业应用即时通信系统怎样和企业已经存在的内部应用系统整合；
5. 企业用户所需求的个人即时通信所不具备的功能。

解决了上述几个问题，基本就满足了一个优秀的EIM的标准，因而本系统致力于以上几个问题的研究与实现[27]，将重点研究企业级即时通信技术所关注的一些热点问题。

本系统订立的目标用户主要是面向中小型企业用户。中小型企业由于人力及财力等因素一般情况下不具备部署例如IBM Lotus Sametime等大型企业级即时通信系统，因而对系统规模小巧，部署简易，功能性强的企业级即时通信系统拥有较大的市场需求。因此本系统订立的原则是针对中小企业用户，建立系统小巧、最终系统部署快捷方便，用户使用简易的企业级即时通信系统[28]。

4.2 系统目标功能描述及问题解决思路

针对上述的问题，本系统的目标即对企业级即时通信系统的功能需求进行分析，讨论其目标要求与技术路线和实现可能，并最终设计实现一套符合企业需求的即时通信软件。

4.2.1 管理性需求

4.2.1.1 权限划分

1. 需求分析

企业员工由于工作职能和职位的不同，在企业内部拥有不同的工作范围和权职。反映在即时通信软件的应用上，即不同的用户可以使用即时通信软件做不同的工作。企业级即时通信系统不能像针对个人用户那样，所有的功能不加任何限制都向用户开放。企业级即时通信系统必须拥有比较完善的用户权限划分能力，使企业级即时通信系统在一个可控制的范围内使用。

在企业需求中，一些用户被控制在一个范围内进行交流，例如只可和所在部门内部的用户进行交流，而其他一些用户可能既可和所在部门的内部用户进行交流，又可以跨部门进行交流，又有一些用户不但可以和企业内部员工交流，也需要和企业外部的其他即时通信系统用户进行交流。甚至一些用户不仅仅具有文本交流的能力，还具有文件发放、召开会议等能力。等等这些类似的问题，其实都是由权限管理机制来进行划分和控制的。可以这样说，权限划分机制是整个企业级即时通信系统最核心的几个功能之一。

2. 目标要求

系统应最终实现用户比较精确的权限划分，让企业内部员工使用本系统具有不同的等级间的区别。并可根据用户的级别维护其管理范围内下属的权限划分，增强企业使用即时通信系统的安全性。

3. 问题解决思路

在 XMPP 协议中并没有用户使用等级的区分，从而也没有权限的有效管理方法，因此我们必须在 XMPP 现有的协议基础上扩展部分结构来支持本系统所需要的权限划分功能。

用户权限的控制在 MIS 等企业应用软件大部分上都是用“组”的方式来

加以控制。这种方式的特点是，授权时方便快捷，可一次大量授权同一权限的用户，减少用户的工作量。但是，在属于同一级别的组中有时需要对个别用户的特别授权，这时候以“组”为授权单位，将产生一定的不便性。因此，本系统中将采用“组”与“个人”的结合方式使用授权，每个用户单独存储其权限控制信息。这样即可达到以“组”授权，方便批量授权用户，又可以实现以“个人”授权的精确控制的目的。

综合考虑本系统的权限划分模块，可以大致分为三类的权限：（1）消息收发的权限管理；（2）功能性应用的权限管理；（3）系统权限分配的管理。

（1）消息收发的权限管理

企业级的即时通信系统用户的最终权限一般都是由管理者在帐号建立之初就已设定好的，本系统也采用这种权限授予方式。综合考虑 XMPP 协议本身所规范的标准以及本系统的需求，出于系统简便高效的目的，消息收发权限的管理方式将采用基于 XMPP 规范标准中的“屏蔽通信”技术实现[24]。

“屏蔽通信”在 XMPP 协议中是由管理某个用户的隐私列表来实现的，通过屏蔽来自某些特定的其他用户的通信来实现屏蔽功能。具体实现为当某个用户提交了该用户的隐私列表至服务器端后，XMPP 服务器将在每次转发该用户的 XML 数据节的时候和该用户隐私列表中的屏蔽条件进行对比，最终做出是否可以放行或截获的动作，达到屏蔽信息的目的。

（2）功能性应用的权限管理

在企业内部使用即时通信工具的员工，由于职位以及职务的不同，其中的一些用户可能需要一些特殊的功能。例如，在一个部门内，部门主管可以给该部门的所有用户广播会议通知；又或者部门主管可以在即时通信工具上召开一个“会议”，邀请相关人员进入参与；或公司文秘发布一些公司的相关文件，供其他用户下载查阅等等，类似这样的功能为了避免不必要的混乱出现和更易于管理的目的将只会赋予企业内部分“特权”员工使用，而大部分的普通员工将不会拥有。因此必须拥有一套“功能性限制”的权限管理体系。

这一部分的权限设定由该用户的管理者统一授权，不受其他用户的干预。实现功能性限制权限管理方式时，系统中的每个用户都必须有自己的独立于其他用户的控制标记。应用端软件则根据这些控制标记，决定是否开放某一项功能给用户。在 XSF 定义的扩展协议中，XEP-0049 私有 XML 数据存储（Private XML Storage）为系统的这一功能性需求提供了技术支持[29]。

XEP-0049 协议定义了每一个拥有 JID 的用户都可以在服务器端存储个人私有数据的协议规范。本系统基于 XEP-0049 协议将每个用户的“功能性限制”控制标记存储于服务端的个人私有数据信息表中。当某一用户登录系统，可以利用<iq/>节获取用户的个人控制标记，来查看应用端软件是否为该用户开放了某一功能的权限。

（3）系统权限分配的管理

系统中将建立这样一种机制，首先由系统管理员统一对一个部门或组别授予权限，在这个部门或组别中将建立个别特权用户，此用户将有管理该部门或组别的权限，提升或降低其管理下的用户权限的功能，提升整个系统的管理效率。

此功能的实现，系统必须扩充系统中的用户信息，以标明在使用管理模块时其管理者的身份，由于在本系统中用户的管理模块独立于即时通信模块，因此标明管理者的身份，可以在系统数据库用户信息中添加相应项作为标识。

4.2.1.2 嵌套组及企业组

1. 需求分析

在企业即时通信系统中，应该具有这样一种机制：在管理者统一建立用户帐号时，可根据用户所在部门的工作性质以及部门，把该用户应归属的组别确定出来，并且设定出该用户可见的其他组别或其他的可见联系人。在该用户首次登陆系统时，终端界面上就已经列出了系统管理员为其设定好的所有初始联系人和初始组别，这些联系人很可能是该用户的领导者、公司内部消息发布者或在工作中需要进行沟通的其他用户。在很多时候，一个用户并不需要把企业中所有的用户都列为他的联系人，只需列出该用户可能需要的联系人即可，这样也避免了过多的非必要联系人给用户终端界面造成混乱，便于管理和控制。

本系统中将这样企业系统管理员提前设定好的组别，称为企业组，以区别用户后期自行添加的个人组用户。企业组的特点是由系统管理员统一建立和分配，且在用户终端上不可以删除、移动、修改该组以及该组内预订的任何联系人，属于一个固定不可变的组别。

在一个上规模的中小企业中，很多时候可能由于用户的精确划分需求，需要将用户的联系人置于多个子组之内。例如一个具有分公司的企业，可能

其设置的根组为各地分公司，在根组之下又划分出各部门。因此嵌套分组的使用在系统中是有必要实现的一个功能。

2. 目标要求

针对上述需求分析，本系统中将实现企业组与个人组的建立，并实现不可删除、移动、修改企业组及组内联系人。实现嵌套组功能的使用。

3. 问题解决思路

（1）企业组与个人组的实现

在实现企业组的不可变的组别性质时，由于每个最终用户所获得的预订的“企业组”可能是不一样的。因此实现企业组的不可变功能时，需要一种机制来标识出“企业组”，使其和其他非企业组有所区分，便于在终端用户界面进行“不可变”的控制。又由于可能用户所获取的企业组是不同的，因此这种特定的标识企业组的方式不能进行全局的定义，只能由每个用户携带各自的企业组的标识。

本系统中将使用 XEP-0049 私有 XML 数据存储（Private XML Storage）扩展协议[29]来标识每个用户的“企业组”标识。该扩展协议为每个用户存储自身特有的数据提供了一种方法。本系统将把标识企业组的标记存储于每个用户的私有数据中，以便从用户下载回来的联系人名册中区分出哪一个是企业组，哪一个是用户组。

（2）名册分组管理及嵌套组的实现

在 XMPP 协议规范中，其本身的名册管理标准，就已经集成了用户组的管理功能。在一个实体终端客户取回他自身的联系人名册时，每一个取回的联系人条目中，都可以包含一个联系人所处的组的信息。

但是在将分组管理引入企业级的即时通信系统后，仅利用 XMPP 协议规范的名册管理标准中的分组功能是远远不够的。且在 XMPP 协议规范中的名册分组管理中只支持一个层次的分组功能，并不支持嵌套组功能。因而 XMPP 协议规范中的分组功能不能满足我们的需求，我们必须对标准协议中的分组管理进行一定程度的扩充以及增加一些权限上控制，才能实现企业级即时通信系统的分组管理要求。

XMPP 扩展协议中的 XEP-0083 嵌套名册组（Nested Roster Groups）协议[30]就是一个利用 XMPP 协议规范中对名册管理标准分组功能进行扩展的一个嵌套分组协议。如果服务端和终端客户端系统都同时实现支持 XEP-0083

扩展协议的话，基于该协议就可以很好的完成上述企业级即时通信系统对于嵌套组的需求。因而，本系统将遵循 XEP-0083 协议的规范实现嵌套名册组的功能。

综上所述，权限划分与名册管理的管理性需求结构如下图 4-1 所示：

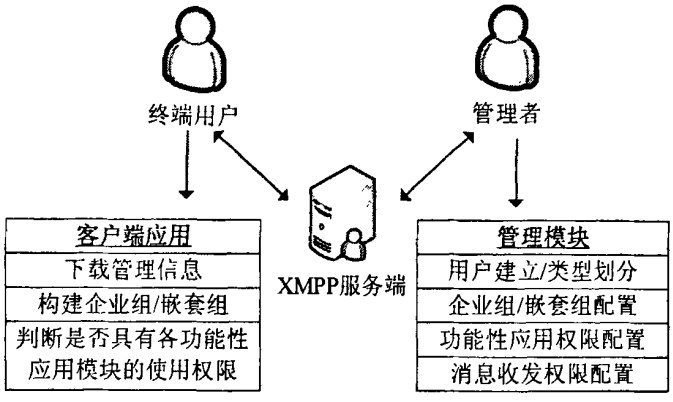


图 4-1 管理性需求分析示意图

4.2.1.3 可追溯性管理

1. 需求分析

企业管理中，在企业内部或与外部客户交流的过程中，某些时候的交流内容可能存在很强的商业利益，涉及到企业的商业机密的安全性；而某些时候也为了避免一些不必要的纠纷，很多时候企业都会将交流内容进行记录。例如企业会议的记录文档、企业客户来访记要和回复记要、以及当前很多的企业的客服电话通话录音等等，这些手段的应用都是为了在企业的管理过程中有一个良好的可追溯性，在需要的时候有据可查。是否具有追溯性管理功能也成为了企业级即时通信系统一个重要的衡量指标，达到痕迹化管理的目的。

2. 目标要求

对用户间的交流内容进行自动记录。当在需要时，可对特定用户的交流内容进行回放。

3. 问题解决思路

在 XMPP 标准协议规范中，用户之间的交流内容都是通过<message/>消息节进行发送和接收的。<message/>消息节中的 to 和 from 属性分别标明了

该<message/>消息节的接受者和发送者，消息节中的<thread/>子元素节指明了一个会话线索，其值是一个在每次会话过程中产生的一个随机的唯一性的字符串，用于一个会话过程的跟踪。当双方用户处于同一个会话过程中时，多次发送与接收到的<message/>消息节中<thread/>子元素的值均相同。在基于 XMPP 协议的即时通信系统中，<message/>消息节都是通过服务器端转发的。因此，为实现可追溯性的功能性需求，我们只需要在服务器端对发出和收到的<message/>消息节和一个发出或收到的时间戳一并进行简单的存储，即可实现可追溯性的功能需求。

这样不对<message/>消息节进行任何的解析而进行直接保存的处理，可使服务器端的处理速度很快，简化了设计实现过程。当企业需要回放记录时，可以通过直接查询数据库中相应的字段，读取其中 XML 数据，根据 to、from、thread，以及时间戳精确的查询到某一用户在特定时间段内和其他用户间的交流内容。

可追溯性结构图如下图 4-2 所示：

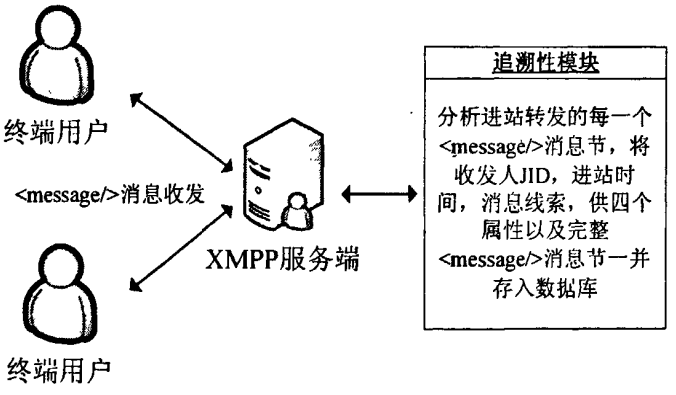


图 4-2 可追溯性分析示意图

4.2.2 功能性需求

针对企业用户在日常办公中的需求，企业级即时通信系统有必要提供一些附加的功能供用户使用，以增强员工使用即时通信系统协同办公的能力。这部分的附加功能，在个人即时通信系统中可能没有提供或者提供了但没有相应可控性。在企业级的应用中，可控性是一项是否符合企业应用的重要参考指标。为了达到这个目标，本系统中的每一个附加功能的实现，将设计为

“可控的”。利用上文讨论的权限分配原则，使每一个附加功能都达到可控制的目的。

本系统将实现以下三项企业日常办公中最为常用的即时通信工具附加功能。

4.2.2.1 会议及讨论组

1. 需求分析

会议功能在即时通信系统中，可以理解为多人参与的交流过程，类似QQ即时通信工具中的“QQ群”功能。本系统将实现两类的群组聊天功能：

（1） 固定的多人群组会话

（2） 临时性的多人群组会话

两者间的不同在于，固定的多人群组会话是一个长期存在的会话过程，需由系统管理员统一建立；临时性的多人群组会话是由某个具有建立临时会话权的用户建立的，当讨论交流结束，多用户会话建立者关闭会话后，该临时性的多人群组会话自动注销。

在本系统中，固定的多人群组会话将设计成“讨论组”，以供企业内部人员进行工作上的一些问题的集体讨论。讨论组一般不设置权限，企业内部的用户可以参与任何一个讨论组，以促进相互学习。临时性的多人群组会话将设计成“会议”系统。在企业内部由于召开远程会议等的需要时，拥有召开“会议”权限的用户将可建立临时性的多人群组会话，在建立“会议”后，邀请相应的与会人员参加，其他没有收到邀请的人员将不能进入该会议室。在“会议”结束后，利用上述的可追溯性管理技术，记录下会议记要。

2. 目标要求

建立一个完善的可控的会议系统功能。

3. 问题解决思路

XMPP标准协议中并没有多人会话的规范。但其扩展协议XEP-0045定义了这一多人会话过程的规范。XEP-0045多用户会话(Multi-User Chat)[31]是XEP中最为复杂的几个扩展协议之一，它详细的描述了多人会话过程的建立、交互过程以及所需要具备的条件，本系统的多人会话过程将在此扩展协议的基础上构建。

4.2.2.2 文件传输与文件分发

1. 需求分析

在企业的日常办公中，相关文件的分发是经常进行的一项工作。应用到即时通信系统中，可以理解为相关文件的一个传输过程。在即时通信系统中，文件的传输功能是一个普遍拥有的功能，XMPP 扩展协议 XEP-0096（File Transfer）[32]也定义了在 XMPP 系统中文件的传输规范。本系统中基于 XEP-0096 文件传输协议实现可控的文件收发功能。

考虑企业的文件分发流程，一般情况下会有专人负责分发相关文件，并不是大部分员工都参与分发过程，相关的文件分发对象只负责接收文件。在以往的即时通信系统中文件分发过程都是点对点的传输，如果分发量较多的话，将会对分发人员产生不必要的工作。因此本系统中考虑使用基于 XEP-0096 文件传输协议一种类似 FTP 文件共享的方式实现文件分发功能。具有分发权限的用户首先将自己需要分发的对象进行一个公示，分发对象看到相应的分发文件公示后，通过一定的机制发送控制命令给分发者，其后自动建立文件传输过程，无需文件分发者的干预。

2. 目标要求

系统建立一套完善的文件传输以及文件分发机制，提升企业用户的工作效率。

3. 问题解决思路

系统中需要实现的文件分发机制，难点在于怎样对分发对象进行所分发文件的公示，以及怎样使分发对象能发送相关的命令通知分发者自动建立文件传输过程。其中涉及到用户间怎么样传输这些控制信息，用户间传输控制信息的方式有多种，最简单的形式就是利用普通的<message/>消息节传输特定格式的控制数据。但作为系统的设计，这样的方式耦合性太高，每一个<message/>的到来我们都需要解析，查看是否含有控制数据，降低了每一种数据的独立性。

综合考虑拟采用 XEP-0047(In-Band Bytestreams)带内字节流传输协议作为控制数据的传输通道。XEP-0047 带内字节流传输协议是一个允许任意两个实体间建立一对一传输字节流的 XMPP 扩展协议，设计的初衷就是为了两实体间传输少量的数据[33]。在此我们用来传输少量的控制数据，是一个非常好

的选择。

4.2.2.3 广播通知

1. 需求分析

企业内部经常有一些组、部门内部或整个企业内部的通知需要发出，例如会议通知、公司聚餐以及公司活动等事宜。应用到即时通信系统上，如果没有“广播通知”的能力，用户间也可以在相应的群组中交流或挨个通知的方法实现，但这样的办公效率是不可接受的。因此本系统中将设计实现一定范围内具有“广播通知”能力的功能，广播范围包括组、部门，以及全公司三个衡量单位。

2. 目标要求

系统实现在特定范围内对所有用户进行广播的能力。

3. 问题解决思路

广播通知实现起来相对容易。在获取用户具有“广播”的特权后，最简单的方式就是遍历所有需要发出通知的用户的 JID 自动向每一个用户发送通知消息。这种方法实现起来也相对容易，在中小企业中，用户数量不多的情况下也比较适宜。此外也有不少的服务端扩展服务实现了上述的广播通知功能，在此我们可以很方便的实现系统的此项功能。

综上所述，系统总的功能性需求有三方面，包括会议及讨论组功能、文件传输与文件分发功能、广播通知功能，结构图如下图 4-3 所示：

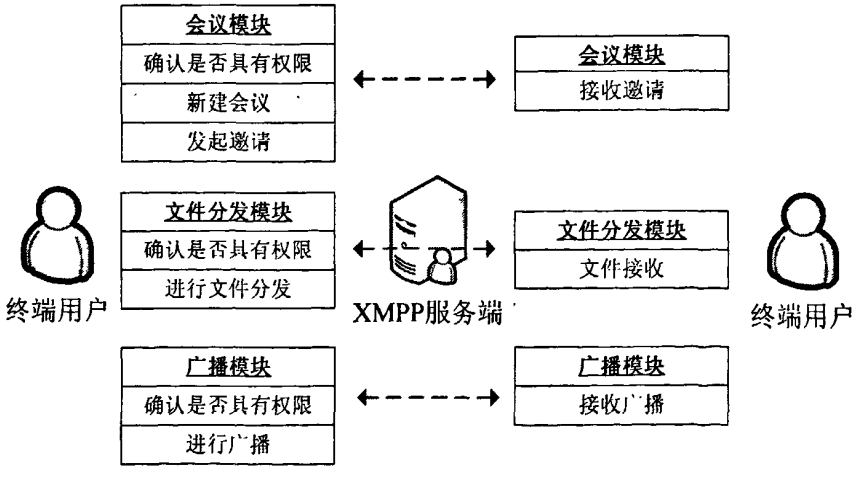


图 4-3 功能性需求分析示意图

4.2.3 扩展性需求

1. 需求分析

在当今企业越来越强调协同化办公、协作化办公的前提背景下，越来越多的公司、企业都有将企业内部的各应用软件相互整合以提高工作效率的要求。XMPP 协议作为一个极易扩展和使用的协议，这一方面的表现是不会让应用该协议的系统失望的。由于 XML 良好的扩展性，将 XMPP 系统的软件与其他软件相互整合是比较容易的。即时通信系统与 MIS、OA 等企业管理、应用软件的相互整合也是企业即时通信系统发展的方向。

本文对扩展性的研究最后给出一个简单的实例，描述 XMPP 协议扩展到其他应用软件系统的一个方法。当然，XMPP 是极易扩展的，甚至可以开发自己的 XEP 协议进行一些更高级别的扩展，本功能的实现仅仅是为了展示基于 XMPP 扩展到其他应用的一个简易例程。该实例的应用可用在诸如从实现了相应接口的 OA 系统上获取该用户的相关任务、作业信息；或通过简单的即时通信消息，将内容发布到企业的网站上等等多方面的应用，可以看出扩展性服务的强大功能。

2. 目标要求

系统实现具有获取另一应用程序实时数据的能力。

3. 问题解决思路

XMPP 系统的扩展能力很强，扩展的方法也很多，通过现有协议和相关的扩展协议的使用可以达到很多的扩展要求，也可以自行设计 XMPP 的扩展协议获取更高级的扩展能力。在本系统中，为获取另一应用程序的数据，将采用基于 XEP-0050 特殊命令（Ad-Hoc Commands）协议[34]的扩展来实现。XEP-0050 扩展协议提供了一整套完善的机制实现与非 XMPP 系统进行交互的能力，通过该协议的应用规范，其他非 XMPP 系统很容易与 XMPP 系统融合。本系统将在非 XMPP 一方的应用程序中建立一个基于 Ad-Hoc Commands 的接口，在 XMPP 系统一方通过发送相关的命令，以便获取非 XMPP 系统的数据。

综上所述，本系统需要实现的功能可划分为两类，一类为基本功能，一类为针对企业级用户所实现的特定的功能需求，总结如表 4-1 与表 4-2 所示：

表 4-1 即时通信基本功能表

序号	功能名称	描述
1	消息收发	用户之间消息的收发
2	名册管理	管理用户帐户下联系人名单
3	出席信息订阅	管理用户向其联系人名册发送其订阅信息以及获取联系人名册的订阅信息

表 4-2 企业级即时通信功能表

序号	功能名称	描述
1	用户管理	包括企业组、个人组的实现，嵌套组的实现
2	权限划分	包括消息收发的权限管理；功能性应用的权限管理；系统权限分配的管理。
3	可追溯性	实现交流记录的自动存储，并可回放
4	会议及讨论组	实现固定的讨论组，并实现召开临时会议
5	文件传输、文件分发	实现文件的传输以及分发
6	广播	向规定范围内广播消息
7	扩展性服务	演示获取另一应用程序实时数据的能力

4.3 本章总结

本章首先对企业级即时通信系统的需求进行了一个概括描述。讨论了企业级即时通信系统在实现过程中需要解决的问题。然后对本系统将要解决的几个关键问题进行了描述，分别就系统的用户管理及权限分配问题、系统的可追溯性管理、其他的功能性需求，以及系统扩展性四个方面阐述了本系统研究的内容、目标以及相应的处理方法和问题的解决思路。

第5章 企业级即时通信系统的设计与实现

5.1 开发平台的选择

XMPP 协议出现至今已有几年的时间，期间得到了各方面的大力支持，出现了一批较完善的基于 XMPP 协议的开源基础库。当今软件工程思想一个重要的观点即是构建可复用的软件。在 XMPP 协议的实现上拥有较为完善的开源基础库的支持下，没有必要再次重复开发，纠缠于底层通信与交互的一些细节，应重点致力于在底层库支持的基础上，通过再次构建、扩展，设计出符合自身要求的软件系统。降低开发成本，提高开发效率，提升新开发软件系统的总体质量，这也正是开源软件出现的初衷与精神所在。本系统的实现过程中将秉承上述原则，利用当前比较成熟和完善的开源库的支持，来构建自身所需的各项应用功能。

5.1.1 服务器端开源软件的选择

Openfire 是一套由 Jive Software 软件组织开发维护的，基于开源软件许可证 GPL 的 XMPP 服务端服务器软件系统。其前身 Wildfire 就已经得到了非常广泛的运用，对 XMPP 标准协议有着完整的支持，是一套完善的 XMPP 服务器端软件。Opernfire 提供了良好的性能以及安全性，它的部署及管理又是非常简便的。Openfire 是使用 Java 语言编写的，因此具有良好跨平台性能[35]。此外，Openfire 在设计之初就已经留出了扩展接口，对于服务端的扩展很容易实现。基于以上因素，本系统将采用 Openfire 作为 XMPP 服务器端软件系统[36][37]。

5.1.2 客户端底层基础库的选择

Gloox 是一套开源的、基于 GPL 开源软件许可证的，使用 C++编写的跨平台的 XMPP 协议客户端基础库[38]。它提供了完善的 XMPP 协议的底层通信支持，封装了 XMPP 核心协议所规定的大部分 XMPP 通信规范，使开发者不用再纠缠于底层通信的细节，可以集中精力致力于上层应用的开发。基于以

上原因，本系统将采用 Gloox 库作为客户端软件开发的基础，在 Gloox 提供的通信支持的基础上完成本系统预订的功能需求开发[39]。

5.2 系统构成

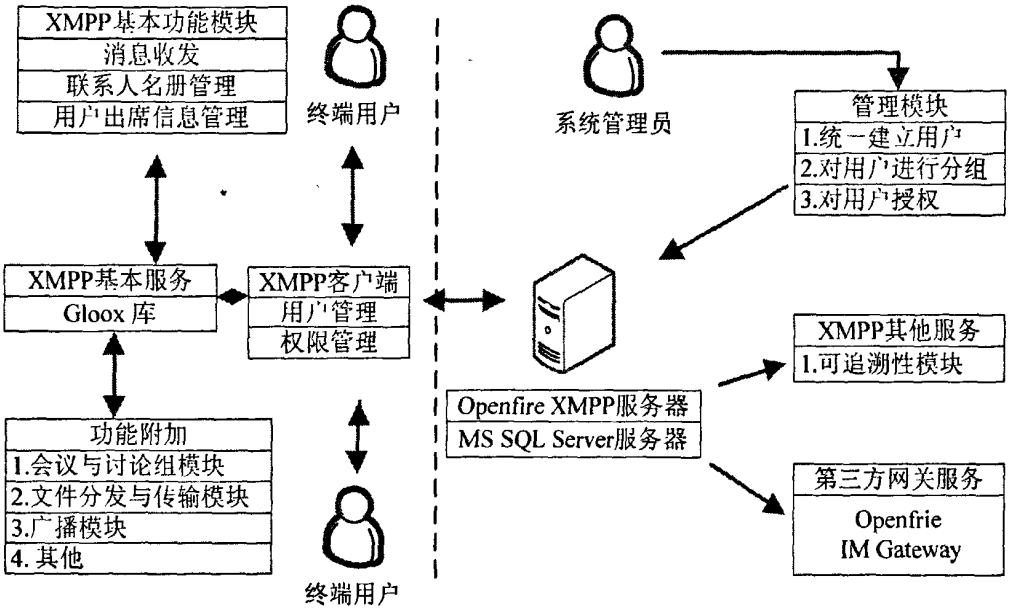


图 5-1 系统构成示意图

如图 5-1 所示，展示了本系统所设计的企业级即时通信系统的基本组成：图示左边的为客户区，右边的为服务区[40][41]。

左侧中，终端用户和 XMPP 客户端直接进行双向通信，XMPP 客户端是基于 Gloox 库构建的。在 Gloox 库提供的底层通信支持下，除 XMPP 标准协议所规范的用户交流、出席信息交互、联系人名册管理等基本功能外，附加功能模块为终端用户提供了丰富、可控的其他功能，包括用户与权限的管理、会议、文件分发、广播等。

右侧中，企业管理员利用管理模块操作 Openfire 服务器和数据库服务器，对用户帐号统一建立、用户的分组和授权进行直接的操作以及建立讨论组，并设计实现一个可追溯性模块作为 Openfire 服务器的插件运行，实现整套系统中消息的痕迹化管理。此外运用了 Openfire 提供的网关 IM Gateway 实现本即时通信系统与其他即时通信系统的互联互通[42]。

5.3 系统数据存储

Openfire 是一套开源的、跨平台的 XMPP 服务端服务器软件系统。对于必要的保证系统能正常运行的系统数据信息以及终端用户信息的存储，Openfire 可采用多套数据库服务器系统为其提供存储支持。例如可使用 MS SQL Server、PostgreSQL、Mysql、Oracle 等等主流数据库服务器系统作为其后台数据库服务器。可以说 Openfire 这一方面提供给用户的可选择性范围很广泛，只要遵行其数据库设计标准，任何一套主流数据库系统都可和 Openfire 相互结合使用。本系统的实现将采用 MSSQL Server 和 PostgreSQL 两套数据库系统分别做为 Openfire 的后台数据库来对系统进行测试和实现。

Openfire 所遵循和使用的数据库结构，核心的关系表如下所述：

1. ofUser 用户表：用于记录系统中终端用户的信息。

表 5-1 ofUser 表

字段	数据类型	含义
Username	nvarchar (64)	用户名
plainPassword	nvarchar (32)	用户密码
Name	nvarchar(100)	用户昵称
Email	nvarchar(100)	用户 email
creationDate	char (100)	创建时间
modificationDate	char (15)	最后修改时间

2. ofRoster 联系人名册表：用于记录系统中终端用户联系人名册信息。

表 5-2 ofRoster 表

字段	数据类型	含义
rosterID	int	联系人名册 ID
Username	nvarchar (64)	用户名
Jid	nvarchar(1024)	联系人 JID
Sub	int	相互订阅状态
Nick	nvarchar (255)	联系人昵称

3. ofRosterGroups 联系人组别表：用于记录联系人所属的组别信息。

表 5-3 ofRosterGroups 表

字段	数据类型	含义
rosterID	int	联系人名册 ID
Rank	int	联系人级别
groupName	nvarchar(255)	联系人所属组名

4. ofPrivate 用户私有数据表：用于记录系统中用户的私有数据信息。

表 5-4 ofPrivate 表

字段	数据类型	含义
username	nvarchar (64)	用户名
name	nvarchar(100)	私有数据名称
namespace	nvarchar(200)	私有数据命名空间
privateData	ntext	私有数据

5. ofPrivacyList 用户隐私表：用于记录系统中用户的屏蔽通信信息。

表 5-5 ofPrivacyList 表

字段	数据类型	含义
username	nvarchar (64)	用户名
name	nvarchar(100)	隐私表名称
isDefault	int	是否是默认表
privateData	ntext	隐私列表数据

以上列举出了 Openfire 所使用的数据库结构表，此外还有一些例如离线信息表等非关键性表，在此不再一一列出。在后续的系统功能实现过程中，也会对上述的基本表进行了一些必要的扩充，在相应的章节将进一步详述。

5.4 XMPP 核心功能的实现

XMPP 核心功能所指的是 RFC3920 以及 RFC3921 所规定的基本的 XMPP 系统最基本的功能：（1）消息收发；（2）出席信息递送；（3）联系人名册管理功能。

本系统中客户端的实现基于 Gloox 的底层通信库的支持。由于 Gloox 库的支持，我们不必去关心底层的 TCP 套接字是怎样去实现连接，怎样收发数据，而致力于关注上层所接收和发送的具体 XML 节[43]。

Gloox 库是一个运用观察者模式编写的 XMPP 客户端支持库，所有的数据交互都是通过“事件驱动”模型完成的。如图 5-2 所示：

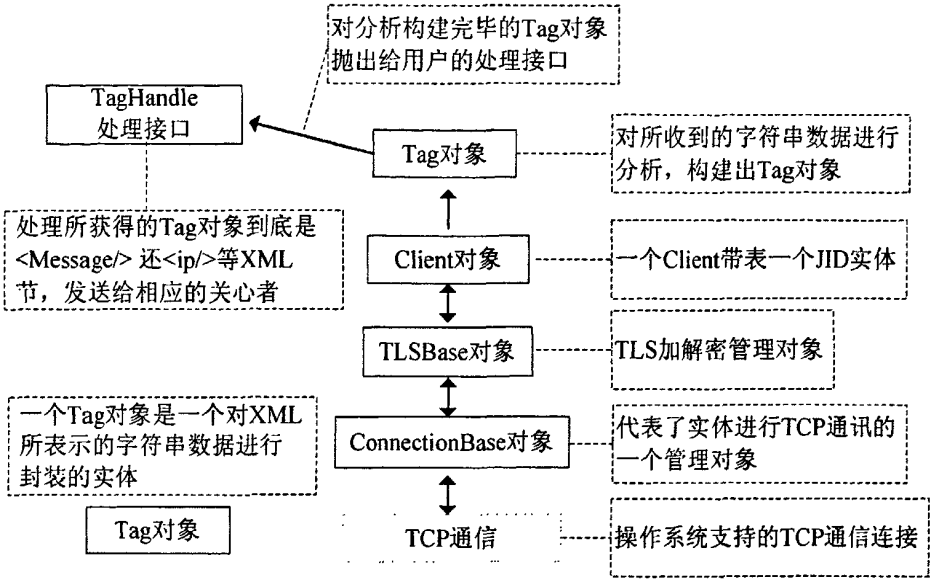


图 5-2 Gloox 实体对象联系示意图

在 Gloox 中拥有几个核心的类：ConnectionBase 类负责与 TCP 套接字的连接通信，在获取 TCP 套接字字符数据后它交与上层的 TLSbase 类进行 TLS 解密工作，解密工作完成后交与 Client 类，该类在 Gloox 就代表一个 JID 实体对象。当 TLSBase 将解密后所收到的字符串交于 Client 处理后，Client 利用 Parser 类将字符串重新解析为一个 Tag 对象，一个 Tag 对象是一个对 XML 所表示的字符串的封装。最后当解析完毕后如果该字符串是一个正确的 XML 元素节的话，它将交于 TagHandle 接口进行处理。我们可以在 TagHandle 接口的实现中，重新分析该 XML 节元素的属性，分析到底属于哪一个类型的 XMPP 节，然后进行相应的处理。当我们需要向另一个实体发送 XMPP 数据时，只需首先构建一个代表该 XMPP 数据节的 Tag 对象，然后交与 Client，Clinet 又用上述相反的顺序将该数据重新进行 TLS 加密，并最后利用 ConnectionBase 将数据发送到服务器端。

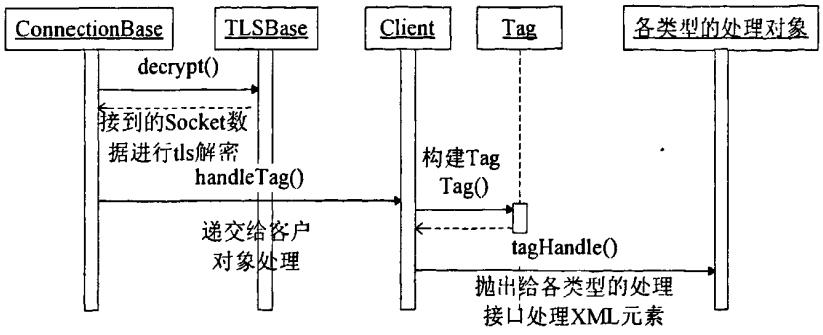


图 5-3 实体接收时序图

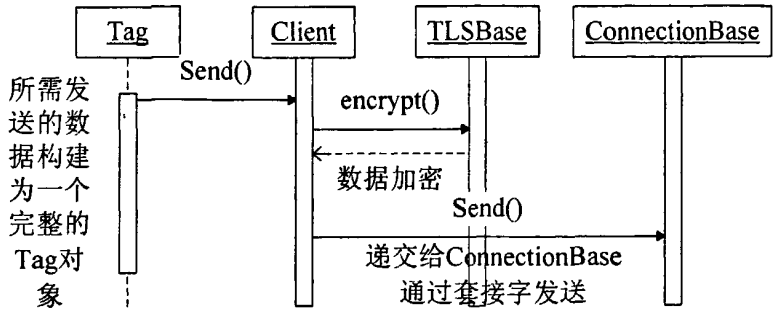


图 5-4 实体发送时序图

在实现 XMPP 所规范的基本功能时，我们只需要构建出一个 Client 对象，并实现相应的 TagHandle 接口，处理我们所关心的<message/>、<presence/>以及<iq/>节即可，将收到的信息交予相应的管理容器处理。当需要回复或请求相应信息时，我们利用 Tag 对象构建相应的回复或请求 XML 节，利用 Client::Send 函数发送至服务端即可完成相应的请求和回复[43]。

在实现 XMPP 的功能需求时，我们需要处理的任何操作以及交互过程，将重点致力于 XML 节元素的解析与生成，在交互双方都可以理解的 XML 节元素的基础上进行交互是非常方便的。

由于 Gloox 库封装了底层的通信过程，我们只需致力于关注 XMPP 协议所规范的 XML 数据节的收发过程即可实现 XMPP 通信。此外 Gloox 也对其他一些常用功能进行了封装，为开发提供了方便。

5.5 功能模块的实现

5.5.1 权限划分

1. 消息收发的权限管理

屏蔽通信是一项 XMPP 标准协议规范中提供的消息筛选机制，遵循该机制的规范，可以达到很好的消息过滤效果。屏蔽通信是通过<iq/>节进行发送的，语法如下：

```
<iq type="set">
    <query xmlns="jabber:iq:privacy">
        <list name="name">
            <item type="[jid|group|subscription]" value="value"
             action ="[allow|deny]" order="unsignedint"/>
                [<message/>|<presence-in/>|<presence-out>|<iq/>]
            </item>
        </list>
    </query>
</iq>
```

例如限制某用户只可在所在的组的范围内进行沟通，不允许与其他外部沟通，其屏蔽通信的<iq/>节如下：

```
<iq type="set">
    <query xmlns="jabber:iq:privacy">
        <list name="example">
            <item type="group" value="mygroup" action="allow" order="1"/>
             <item action="deny" order="2"/>
        </list>
    </query>
</iq>
```

上例中两个<item/>元素分别标识了两种类型的屏蔽。第一个<item/>类型为“group”值为“mygroup”动作为“allow”，标明了允许和该用户联系人名册中任何一个“mygroup”可以相互交流和发送出席信息。第二个<item/>

没有类型说明，也没有键值，这种情况代表所有的类型和值，动作为“deny”，标明拒绝所有的交互。两者组合标明，除了可以和 mygroup 组里的成员可以进行交流外，拒绝其他所有交流。

通过上例可以看出，XMPP 标准协议规范的屏蔽通信的功能是非常强大的，利用好屏蔽功能，可以达到很好的可控性能。通过系统管理员工具直接操作 Openfire 用户数据库，对相应用户的屏蔽 XML 数据信息直接写入数据库实现。

2. 功能性应用权限管理

本系统中所有的附加功能和部分的即时通信工具基本功能，都要在可控制的范围内使用。因此，权限的划分极为重要。功能性限制的权限管理的处理上将采用 XEP-0045 私有数据存储扩展协议来存储每一个用户拥有的权限信息。设计权限表 XML 为：

```
<authority>
    <item name="AddNewUser"  value="true"  scope="none">
    <item name="CreateConference"  value="ture"  scope="ALL">
    <item name="FileDistribution"  value="true"  scope="ALL">
    <item name="Broadcast"  value="true"  scope="ALL">
    <item name="FileTranslate"  value="true"  scope="ALL">
</authority>
```

权限表每一个<item/>节依次标识的信息为：是否可以自行添加联系人；是否可以发起会议；是否可以进行文件分发；是否具有广播能力；是否可以传输文件。value 值为 true 标识具有这个权限，false 为没有。

scope 标识具有该项权限的范围。取值为：none－无区分；ALL－所有区域；domain：所在域内；Group－所在根组内；GroupLevel1－所在嵌套 1 级组内；GroupLevel2－所在嵌套 2 级组内。取值依据以上顺序拥有这个权限的范围依次缩小。特别说明：如果 scope 取值为 domain 范围之下，所有的 Group 值都只针对该用户所处的“企业组”内有效，对用户自行添加的联系人都无此项权限。

当用户登录后，首先获取该用户的私有数据，确定该用户拥有的权限。用户与服务器端的交互 XML 如下：

C 端:

```
<iq type="get" id="get authority _1">
    <query xmlns="jabber:iq:private">
        <authority xmlns="example/authority"/>
    </query>
</iq>
```

S 端:

```
<iq type="result" from="local@example.com/resource"
 to="local@example.com/resource" id="get authority _1">
    <query xmlns="jabber:iq:private">
        <authority xmlns="example/authority">
            <item name="AddNewUser" value="1" scope="none">
            ……
        </authority>
    </query>
</iq>
```

当客户端获取自身的权限文件后，根据上述的权限表，客户终端加以相应的限制，实现可控的目的。本系统的权限设置将由系统管理员通过管理工具在新建用户帐号时，由系统管理员直接操作 Openfire 用户数据库，对相应项直接写入 XML 权限表实现。

3. 系统权限的分配管理

为了实现分级管理的目的，本系统将对 Openfire 数据库的用户信息表进行了扩充，增加标识用户级别的字段，拟定义四种用户级别：系统管理员——具有完全的管理运用能力；部门管理员——对该部门内所有用户具有管理能力；组管理员——对该所属组具有管理应用能力。普通用户——不具备任何管理能力。每一级别的用户将具有相应的管理权限，可对所管辖的人员权限进行调整，但不可调整为高于系统管理员对改组设置的最高权限。对 Openfrie 数据库中 ofUser 表中进行扩充，增加所需要的属性存储列。增加 UserLevel 列，描述了用户所属级别，字段类型为 int。

5.5.2 嵌套组及企业组

1. 嵌套分组实现

用户分组功能在许多即时通信工具都有实现，但是具有嵌套组的实现在大部分的即时通信工具中都没有实现，XMPP 也不例外。不过 XMPP 协议中的扩展协议 XEP-0083 却给出了使用嵌套分组的规范。XEP-0083 扩展协议对嵌套分组的实现是和另一个扩展协议 XEP-0049(Private XML Storage)私有数据存储相互配合实现的。

在 XMPP 协议规范中用户的联系人名册中某一个用户所属的组是用<group/>子元素在获取联系人名册中的相应<item/>元素下标识的。例如：一个用户请求他的联系人名册和服务端的交互 XML 如下：

C 端发送：

```
<iq from="local@example.com/resource" type ="get" id="roster_1">
    <query xmlns="jabber:iq:roster"/>
</iq>
```

S 端返回：

```
<iq to="local@example.com/resource" type ="result" id="roster_1">
    <query xmlns="jabber:iq:roster">
        <item jid="remote@example.com" name="remote"
         subscription ="both">
            <group>MyFriends</group>
        </item>
        ……
    </query>
</iq>
```

在 S 端的返回值中，每一个<item/>元素标明了一个联系人，其中<group/>标识了该联系人所处的组别。上例中 remote@example.com 这个联系人所在的组就为 MyFriends 组。

XEP-0083 扩展协议中的规范是使用一个定界符在<group/>值中标明嵌套组的层次。例如<group>MyFriends::BestFriends</group>使用了“::”作为定界符标明了该用户所属的组别为 MyFriends 组之下的子组 BestFriends，形

成了一个嵌套组。服务端使用 XEP-0049 私有数据存储，把定界符存储在每个用户的私有数据中返回给客户端。客户端再根据返回的定界符在<group/>值中确定出嵌套组的层次。本系统将使用 XEP-0083 所推荐的“::”作为定界符处理用户的嵌套组。

由于 XEP-0083 协议并没有被本系统采用的开源软件支持，在实现中本系统将模拟 XEP-0083 协议所规范的嵌套组来处理用户所属的嵌套组问题。强制<group/>元素的值中只要出现“::”元素即是处于嵌套组中。由于嵌套组不可能无限制的嵌套，且嵌套组过多将在客户端的系统上表现混乱，所以本系统的实现中将只处理三层的嵌套组。实现流程如图 5-5 所示：

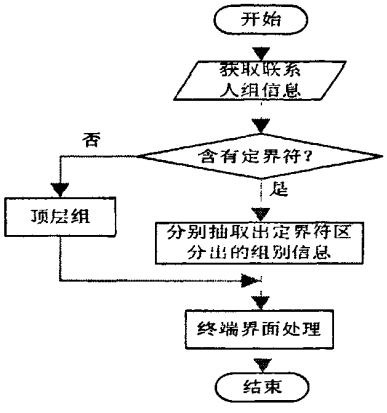


图 5-5 嵌套组处理

2. 企业组的实现

企业组功能的实现关键在于怎样让客户端区分出企业组和个人组。本系统将采用 XEP-0045 私有数据存储扩展协议来存储每个用户“企业组”的标识。设计企业组标识 XML 为：

```
<EnterpriseGroup>
    <item name="GroupName"/>
< EnterpriseGroup/>
```

每一个<item/>节 name 属性所标识的值就为该用户所获取的企业组信息。当用户登陆后首先获取该用户的私有数据，获取其企业组的标识，当下载回来联系人名册后，对比标识中的企业组名称，如果为企业组，则设定该组为“不可变”的。

在系统管理员统一建立系统内用户帐号时，有必要在用户的基础资料中

加入用户所属部门的属性。该属性的加入有利于企业对人员的组织，也有利于系统管理员对用户的管理。因此，我们需要对 Openfire 数据库人员管理表 ofUser 表中进行扩充，增加所需要的属性存储列。增加 department 列，描述了其人员所属部门，字段类型为 nvarchar(100)。

5.5.3 可追溯性管理

为了实现可追溯性，达到良好的包监听并保存的目的，应该在服务器端实现相应的包监听接口，在每监测到一个<message/>信息节就将其保存在数据库中，以便事后回放需要。

服务器端软件系统 Openfire 为用户提供了方便的接口扩展的服务。Openfire 以插件的形式来扩展其服务，在符合它插件包的条件下，实现其提供的 Plugin 接口下的 InitializePlugin 方法以及 destroyPlugin 方法即可实现插件的加载和使用[44]。

本功能是在实现了上述接口的前提下，创建一个<message/>包的监听，记录下<message/>包的内容和时间戳一并写入数据库，以便需要时从数据库中获取并回放。在 Openfire 中为我们提供了 PacketRouter 接口，用于对所有进入 Openfire 的 XML 数据包的路由。重写 PacketRouter 接口下 route(Message packet)方法即可实现所需要的功能[45][46]。部分实现代码如下：

```
public class HandlerMessagePlugin implements Plugin, PacketRouter
{
    public void destroyPlugin()
    {
    }
     public void initializePlugin(PluginManager manager, File pluginDirectory)
    {
        //进行必要的连接数据库初始化，以及变量初始化
    }
    public void route (Message packet)
    {
        SimpleDateFormat time   =
        new SimpleDateFormat("yyyy-MM-dd-HH-mm-ss");
```

```
            String strDatetime = SimpleDateFormat.toLocalizedPattern();
            String strPacket = packet.toXML();
            String strThreadID = packet.getThread();
            String strTo = packet.getTo();
            String strFrom = packet.getFrom();
            //.....数据库操作，五个字段一并存入数据库。
        }
    }
```

在监听到了所需要的包后，取当前时间日期，和当前包的线索 ID，以及收发人 JID 一并存入数据库。收发人 JID 以及线索 ID 的存入是为了回放时的检索方便。

在 Openfire 数据库端新增 MessageLog 记录表，用来存储<message/>数据包[47][48]。

表 5-6 MessageLog 表

字段	数据类型	含义
MessageTo	nvarchar(100)	接受者 JID
MessageFrom	nvarchar(100)	发送者 JID
CreateDatatime	datetime	生成时间
ThreadID	nvarchar(100)	线索 ID
Xml	ntext	交流全文

5.5.4 会议及讨论组

在 XMPP 扩展协议 XEP-0045 中具有会议功能的完整实现规范。Gloox 库中已经包含了 XEP-0045 协议的部分实现。本系统的实现中，会议功能是与权限分配相关联的。在会议发起者建立会议室后，会议发起者邀请相应需要参与的人员参加，对于其他人该会议室是不可见的，保证了会议的召开不受其他人的干预。

在 Gloox 中 Client 类代表一个 XMPP 的终端实体，MUC-InvitationHandler 类是一个会议邀请接口，如果有其他用户邀请该用户加入某个会议，该接口中的方法 handleMUCInvitation 将被触发，即可知道是由谁在邀请，参加的是

什么会议。因此我们只需将 MUCInvitationHandler 接口在 Client 类的具体实现中注册，并实现该接口就可以达到监听是否有其他用户邀请参加会议的功能，然后做出相应的处理。部分实现代码如下：

```
class ConferenceInvite: public MUCRoomHandler
{
    public:
    ConferenceInvite (Client *client)
    {
        client->registerMUCInvitationHandler(this);
    }
    virtual void handleMUCInvitation( const JID& room,
    const JID& invitee, const std::string& reason,
    const std::string& body, const std::string& password, bool cont )
    {
        //将会议邀请人“invitee”，邀请原因 “reason”，以及会议
        //主题“body”反馈给用户，让用户判断是否参与会议。
    }
}
```

在实现上述的会议邀请监听实现后，如有需求我们即可参与该会议。参与会议部分代码如下，新建会议代码与参与会议的不同之处在于，如果该会议室已经存在则加入该会议室，如果不存在则新建会议室。

```
class Joinconference
{
    public:
    Joinconference(Client *client, const std::string& roomname,
    const std::string& nick, const std::string& domain)
    {
        std::string strNick = roomname + “@” + domain + “/” + nick;
        JID nick(strNick);
        m_room = new MUCRoom(client, nick, this);
    }
```

```
    void Join()
    {
        m_room->join();
    }
    .....
    public:
    MUCRoom *m_room;
}
```

在XMPP系统中如果服务器端提供了XMPP标准协议规范以外的功能，客户端都是由 XEP-0030 (Service Discovery) 服务发现扩展协议[49]来发现服务的。例如实现了多用户聊天功能的服务器，通过服务发现可以获得所有房间的信息。因此在会议系统中需要设置获取的房间信息，使其他与会议不相干的人员对会议房间不可见。因为本系统中还提供公用的讨论组给所有的用户使用，因此可以根据从服务器端获取的房间信息来判断是否可见。公用讨论室的房间都是由系统管理员直接在Openfire服务端添加设置的拥有可持久性属性的房间，而会议是用户在是使用过程中临时创建的，因此房间不具有持久性。在从服务器端获取房间的属性时，有一项XML数据块为：

```
<field var="muc#roomconfig_persistentroom" >
<value> 0</value><field/>
```

标识的即为房间是否为固定的持久的房间。在客户端获取房间属性时，如果房间不为持久性质的，将在客户端不可见，达到了隐藏会议房间的要求。

当用户需要查看哪一个讨论组正在开放时，利用上述提及的 XEP-0030 服务发现扩展协议即可发现该讨论组，在获取房间的信息后使用上述加入“会议”的功能代码既可实现加入讨论组的功能。

5.5.5 文件传输与文件分发

文件分发功能的重点在于建立一种机制，当文件分发者把需要发送的文件的信息提供给接收者后，接收者如需下载所需的文件，通过一种机制来通知文件分发者建立文件发送通道，开始发送文件。

本系统将通过XEP-0047 In-Band Bytestreams 带内字节流协议规范来传送文件分发的控制信息。带内字节流协议适用于两终端实体间发送少量字节

流数据。文件分发控制表如下：

<FileList>

<item name="name" description="description"

action=[set|get|ok|error] add="JID"/>

<FileList/>

文件分发者首先设置上述 XML 文件分发控制表 name 属性标识需要分发的文件名称；description 属性标识文件的简要说明；action 属性标明此 item 的动作，如为 set 标明是文件分发示意过程，get 标明文件获取过程，ok 标明文件分发准备好过程，error 标明错误过程；add 为文件分发接受者的 JID。流程图如图 5-6 所示：

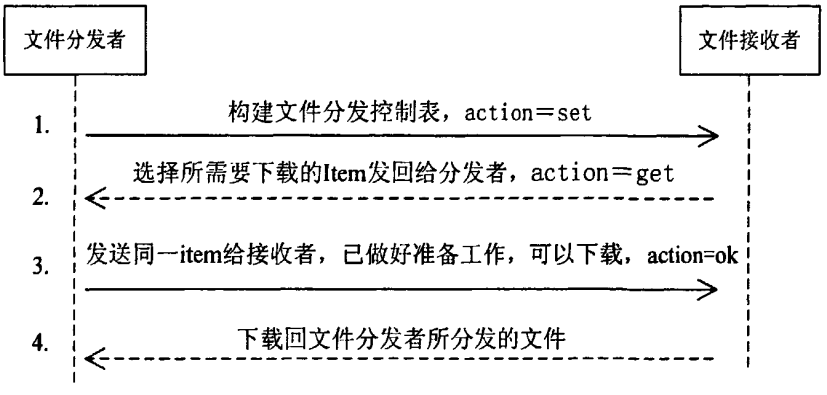


图 5-6 文件分发控制图

当文件分发者接收到来自分发对象的“get”<item/>后，首先查询该文件是否还在分发有效期内，如果还在有效期内，文件分发者将准备文件的发送过程完成后，返回分发对象“ok”的同一<item/>，文件分发者即可下载回所需文件。若返回“error”<item>则标明，该文件分发已经过期，不再有效。

文件的传输在 Gloox 库中提供了 SOCKS5BytestreamSOCKS5，BytestreamDataHandler 等类实现了基于 XEP-0065 协议[50]在此扩展的 XEP-0096 文件传输协议，基于 Gloox 提供的接口类就可实现接收和发送文件数据。部分代码如下：

文件发送：

```
class FTSend: SIProfileFTHandler, SOCKS5BytestreamDataHandler
{
```

```
    void start()
    {
      Client* j = new Client(..);
      //注册一个用来发送文件的服务
      m_server = new SOCKS5BytestreamServer(j->logInstance(), 6666 );
      //生成一个文件传输管理对象
      SIProfileFT* f = new SIProfileFT( j, this );
      f->registerSOCKS5BytestreamServer(m_server);
      //设置文件中转服务器
      f->addStreamHost(j->jid(), "example.com", 6666 );
      //请求发送一个文件
      f->requestFT(to, file, size );
     }
     virtual void handleFTSOCKS5Bytestream (SOCKS5Bytestream* s5b )
     {
      //请求得到回复，发送文件
      if(s5b->connect())
      {
        s5b->send("everything..");
      }
    }
};
```

文件接收：

```
class FTRecv : SIProfileFTHandler, SOCKS5BytestreamDataHandler
{
  public:
    void start()
    {
      Client *j = new Client(...);
      SIProfileFT *f = new SIProfileFT( j, this );
      //设置文件中转服务器
```

```
        f->addStreamHost(j->jid(), "example.com", 6666 );
    }
    virtual void handleFTRequest(....)
    {
        //接收到文件发送者发送文件请求，同意并受理
        f->acceptFT( from, id, SIProfileFT::FTTypeS5B );
    }
    virtual void handleSOCKS5Data( SOCKS5Bytestream* s5b, const std::string& data )
    {
        //接收到数据 data 进行存储。
    }
};
```

5.5.6 广播通知

广播模块实现相对简单，在用户登录时获取了个人权限表后，根据所拥有的权限，在自身所在子组、组、或整个企业内部，枚举所有用户，依次发送消息即可。

本系统中使用 Openfire 提供的一个“Broadcast”插件实现广播功能。该插件是作为一项其他服务存在于 XMPP 服务器端，因此该插件拥有一个 JID，例如@broadcast.example.com 用于标识该服务，该方法只需设置该 JID 的 Node 部分为所需发送的目的地址即可，例如 all@broadcast.example.com 或 mygroup@broadcast.example.com，实现的效果分别为：向所有人发送广播；向 mygroup 组发送广播。

5.5.7 扩展性服务

本系统的扩展性服务模块，只是作为一个演示的目的出现在系统中。系统将与一非 XMPP 系统程序进行通信，获取该程序上的实时数据，以此演示 XMPP 系统扩展到其他系统并与其交互的能力。

本实验将建立一个 Windows 程序，来模拟系统外的另一程序。该程序将简单的设置两文本框，每当 XMPP 系统发送特殊命令传送到该程序一端时，

该程序将动态的返回两个文本框中的实时数据。该 Windows 程序实现了一个 XMPP 处理接口，因此它也具有一个 JID 用于标识自身，其他实体正是通过这个 JID 来与它交互的。

为了实现交互能力，系统中使用了 XEP-0050（Ad-Hoc Commands）扩展协议。该协议适用于两实体间，其中一方发送特殊命令至另一方，当另一方获取命令后进行特定数据的返回，以此进行数据交互。在使用 XEP-0050 前，必须设定一组双方都可以理解的命令格式。设计的 XML 流交互如下：

XMPP 系统：

```
<iq type="set" to="remote@example.com" id="exec1" >
    <command xmlns="http://jabber.org/protocol/commands"
    node="getinfo" action="execute"/>
</iq>
```

非 XMPP 系统：

```
<iq type="result" from="remote@example.com"
to="local@example.com" id="exec1" >
    <command xmlns="http://jabber.org/protocol/commands"
     node= "getinfo" status="completed">
        <x xmlns="jabber:x:data" type="result">
            <item value="abc">text1<item/>
            <item value="def">text2<item/>
        </x>
    </command>
</iq>
```

如上 XML 流交互所示，当 XMPP 系统送了一个“getinfo”并且 action=“execute”的 Command 命令给非 XMPP 一方后，表明了 XMPP 系统将要获取规定的数据。当非 XMPP 一方接收到该命令后，首先确认命令的种类，当和预订的“getinfo”类型相同时候，即获取两文本框的内容，然后构建 XML，填入<item/>元素的 value 属性中，其中<item/>元素值代表是哪一个文本框。完成一次信息的交互。流程图如图 5-7 所示：

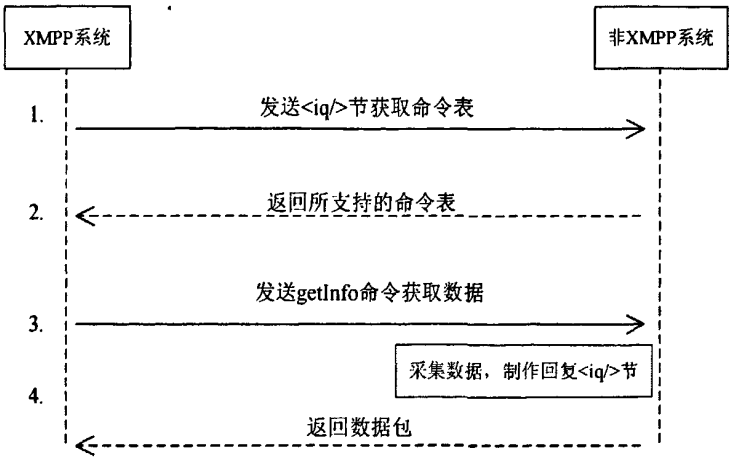


图 5-7 与非 XMPP 系统的信息交互

5.6 本章总结

本章中对企业级即时通信系统的几个关键技术进行了实现的说明阐述。分别就对权限划分、嵌套组与企业组、可追溯性管理、会议以及讨论组、文件传输与文件分发、广播通知等功能模块的实现进行了阐述和流程的示意。本章最后一节扩展性服务中给出了XMPP系统与非XMPP系统之间的简单交互方法，模拟了 XMPP 协议扩展到其他系统或和其他系统相结合的能力。

第 6 章 系统测试

系统测试是系统研发过程中质量控制以及验证系统设计是否符合需求的一个重要手段。本章将对系统进行一个比较全面的测试，以检验系统是否已经符合企业级即时通信系统的需求标准。

6.1 系统测试环境

由于系统测试条件有限，不具备进行大规模应用以及性能测试条件，因此系统测试将重点致力于系统的功能性要求是否符合需求标准以及功能的完整性测试。测试环境如图 6-1 所示：

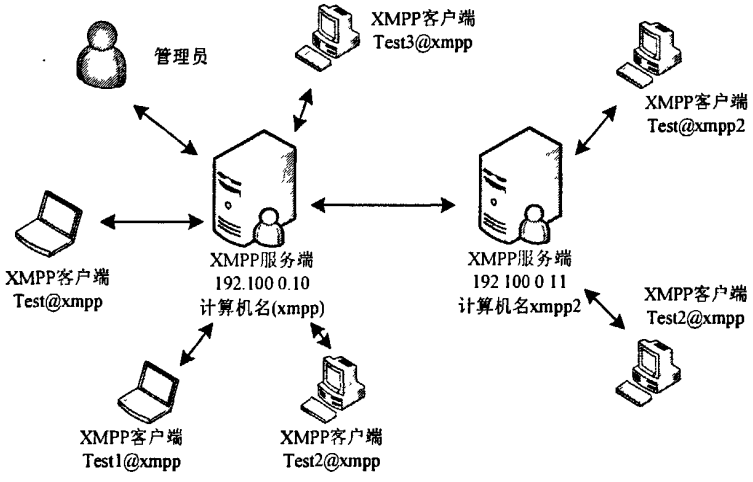


图 6-1 系统测试环境

- **硬件环境：**

（1）服务器两台，系统分别为 Windows 2000 Server 和 Windows 2003 Server，分别装有 Openfire XMPP 服务器，数据库服务器分别为 MS SQL Server 2000 与 PostgreSQL 8.3。

（2）测试 PC 机 6 台，系统分别为 Windows XP 以及 Windows2000。

测试系统中建有两台 XMPP 服务器，计算机名称分别为 xmpp，和 xmpp2，分别连接有六台客户终端，管理员利用管理工具连接 XMPP 服务器，对系统内部用户首先进行帐号建立和权限配置。

6.2 系统测试范围

本章所进行的测试根据其类型可以分为两部分：

（1）XMPP 核心协议规范的即时通信系统应具有的功能是否完整。包括用户联系人名册的获取，出席信息的交互，以及消息的收发过程。

（2）本系统所设计的企业级运用是否完整。包括用户的管理、用户权限设置、文件分发、会议系统、广播等。

6.3 系统测试过程

6.3.1 用户帐号建立以及权限设置

如图 6-2 至 6-4，系统管理员首先建立用户帐号，而后为用户分配权限，设置通信范围，以及设置该用户的企业组信息。

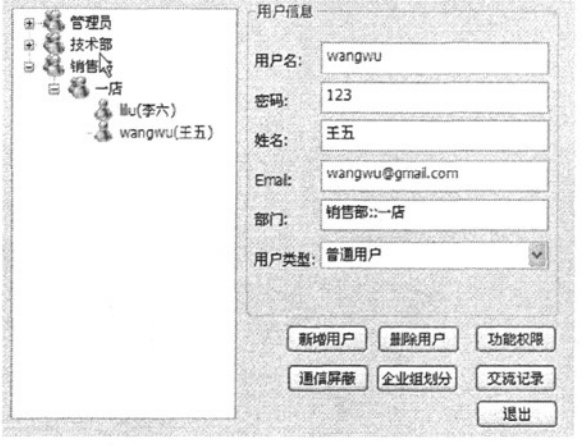


图 6-2 建立用户帐号

图 6-3 用户权限设置

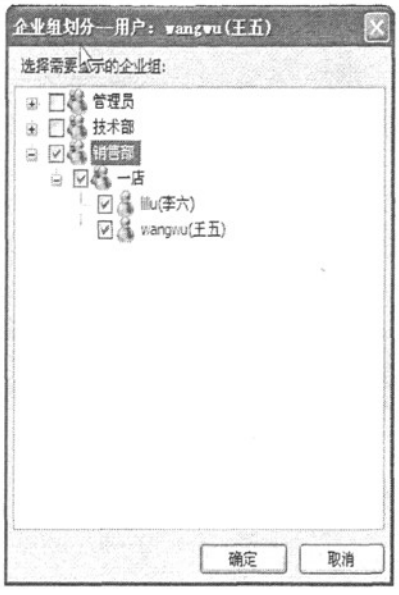


图 6-4 企业组设置

6.3.2 XMPP 协议规范基本功能

图 6-5 所示，是本系统的基本功能实现。描述了用户“wangwu”的登陆以及获取联系人列表后界面截图。图中可以看出刚我们为其设置的企业组，以红色显示联系人。

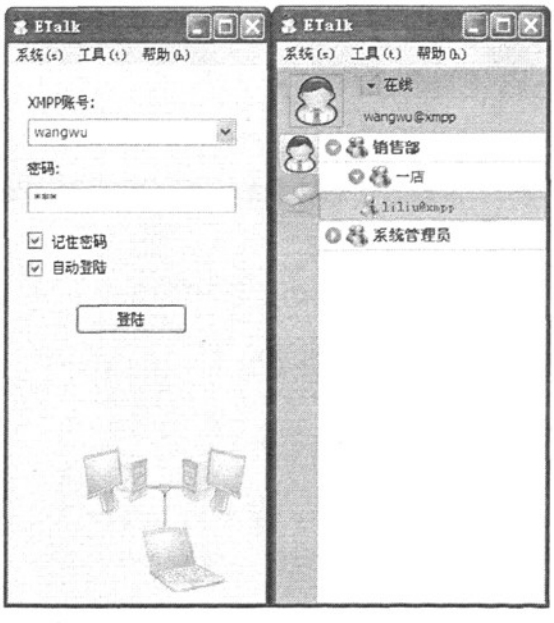


图 6-5 用户登录及联系人列表获取

图 6-6 所示，描述了用户“wangwu”与“liliu@xmpp”用户之间的交流过程。

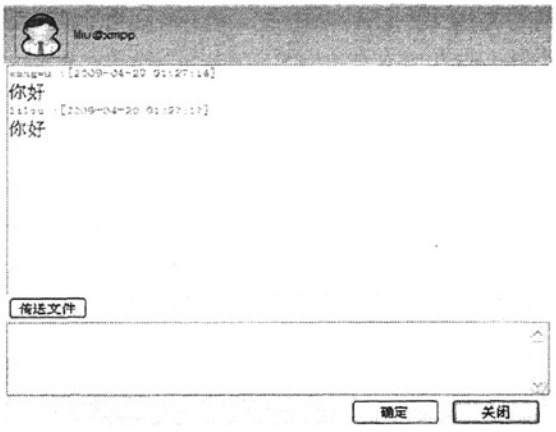


图 6-6 用户交流

6.3.3 其他功能实现

- **会议功能**

如下图 6-7 描述了用户“wangwu”在拥有“会议”创建权限后创建会议，如图中“1”所示，并邀请用户“liliu@xmpp”参与如图中“2”所示。

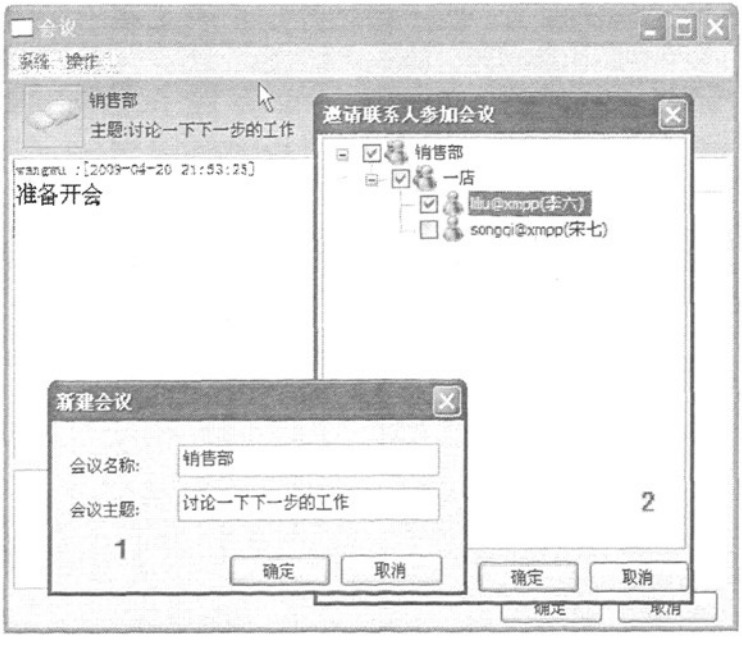


图 6-7 会议功能

当用户“liliu@xmpp”收到邀请后，即可加入会议实现多人会议功能。

- 文件分发

如下图 6-8、6-9 所示，描述了拥有文件分发权限的用户“wangwu@xmpp”向用户“liliu@xmpp”和“songqi@xmpp”分发文件的过程。

用户“wangwu@xmpp”首先添加了需要分发的文件和接收用户 JID，进行文件的分发，当点击分发后用户“liliu@xmpp” 和“songqi@xmpp”将跳出如图 6-9 的界面，提示用户下载分发的文件。

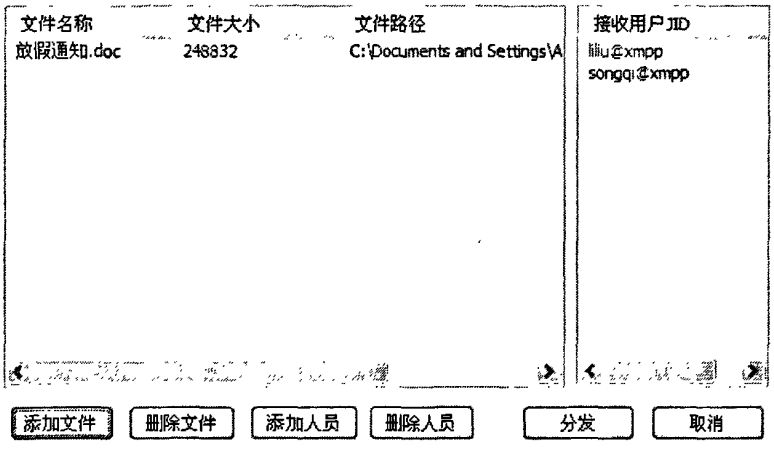


图 6-8 文件分发过程

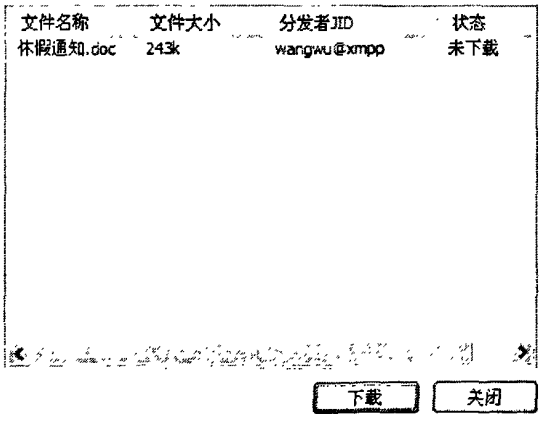


图 6-9 文件接收

- 系统广播

图 6-10 展示了拥有广播权限的“wangwu@xmpp”用户，像所在部门内进行广播的示意图。

图 6-10 展示了用户“liliu@xmpp”接收到用户“wangwu@xmpp”的广播信息。

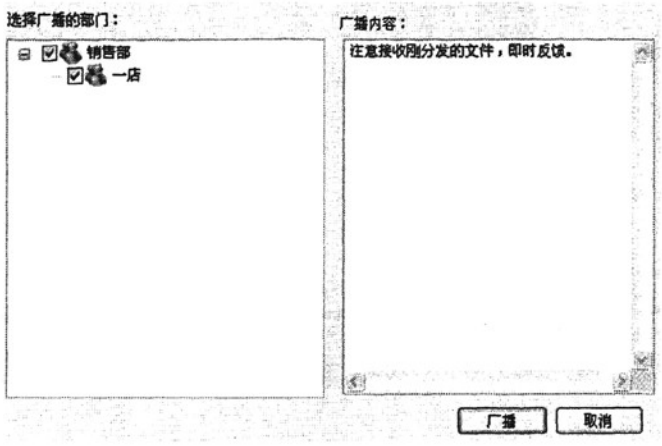


图 6-10 系统广播

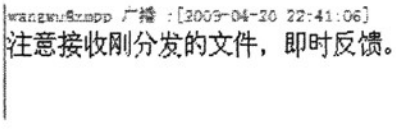


图 6-11 用户接收到的广播信息

- **扩展性支持**

图 6-12 展示了本系统从非 XMPP 的另一程序上获取了界面上两编辑框控件上的事实数据，展示了 XMPP 系统和其他系统的一个交互过程。

其中图 6-12 由三张截图合成，窗口 1 表示非 XMPP 的另一应用程序，窗口 2 为本系统，窗口 3 为点击“扩展性测试(t)”菜单后从非 XMPP 系统返回的实际数据。

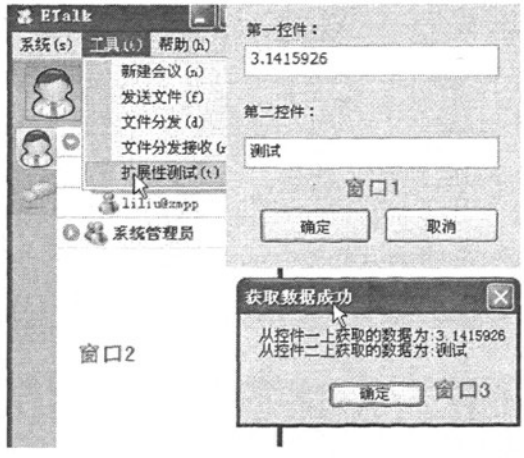


图 6-12 扩展性测试

6.4 本章总结

本章对系统各功能的功能性要求以及完整性要求进行了比较详细的测试，依次分别就各单元模块单独展开测试，搭建了测试平台，建立测试用例，测试是否达到了预期的系统要求。并列出了各个功能模块的测试截图，显示了测试结果。系统实验测试的结果表明，系统功能完善、稳定，界面友好简洁，满足企业级即时通信系统的需求。

总结与展望

1 总结

XMPP 即时通信与出席协议由于其自身的开放性、安全性、简易性，及易扩展性等方面的优势，在即时通信市场中成为一种较为活跃的即时通信技术，Google 的加入更是确定了其在即时通信技术方面的主流地位。XMPP 即时通信协议技术如今已经不再局限于 IETF 所规范的两个文档，更多的研究与应用体现在对 XMPP 扩展协议的研究与应用之上。基于 XMPP 协议的各方面表现出来的优势，本文设计了一套基于 XMPP 协议及其扩展协议的企业级即时通信系统的解决方案并加以实现。针对目前即时通信系统应用到企业日常办公中所暴露出来的不足，分别就用户的可管理性、安全性、企业应用的功能性要求以及和其他非 XMPP 系统的交互整合，设计了技术解决方案并加以实现。基于 XMPP 技术将是即时通信技术的一种未来趋势，因此开发出一套符合企业需求的即时通信系统未来将会有很大的市场空间。

本文主要的研究结果如下：

1. 对目前市场上存在的主流即时通信技术、协议进行了分析比较，对 XMPP 协议及其扩展协议进行了深入的分析，对后续的设计及实现奠定了理论基础。分析了目前即时通信系统在企业中应用的不足，以及 XMPP 协议运用于企业所体现出来的优势。

2. 针对企业级即时通信的应用需求，分别就用户管理、权限分配、可追溯性等管理性需求；会议、文件分发、广播等功能性需求；以及扩展性服务需求，提出了企业化管理中的解决方案并加以实现，达到企业级即时通信系统的要求。系统还模拟了 XMPP 系统与其他非 XMPP 系统之间进行交互的过程，展示了 XMPP 系统与非 XMPP 系统相互整合与扩展的能力。

2 展望

XMPP 协议的特点决定了基于 XMPP 协议的企业级即时通信系统在不远的将来会有很大的发展空间，研究与实现基于 XMPP 协议的即时通信系统未

来将有很高的应用价值与经济价值。在本系统的设计和实现过程中，由于个人时间和精力有限，许多实用性较高的功能都没有添加入系统内，许多地方也有待于进一步提高与完善，存在很多不足之处。下一步的工作方向是将一些实用性较高的功能逐步加入到系统之中。例如白板功能、语音以及视频支持，远程协助支持等；以及和企业现有系统应用的相互整合，例如设计出完善的接口，使即时通信系统与企业的应用系统 OA、MIS、CRM 等的相互整合，真正发挥出协同办公的优势。

致 谢

本文是在我的导师楼新远老师悉心的指导下完成的，从收集资料、论文选题以及在论文的撰写过程中，楼老师都给了我耐心的指导。在这三年的学习生涯中，楼老师不仅给我提供了很好的学习环境，而且无论是在学习还是在生活中，楼老师都给予了我极大的帮助。在此表示衷心的感谢，而其深厚的学术素养和高尚的品格将永远是我为人处事的楷模。

感谢我实验室的同学，与你们在学术上的讨论，让我受益匪浅！

感谢我的家人与男友，一直以来给我的关怀和支持，使我得以顺利完成学业！

最后，感谢在百忙之中抽出宝贵时间参加我的论文答辩的各位评委老师，您们宝贵的意见与建议将给我以后的科研工作以启迪和帮助。

参考文献

[1] 艾瑞市场咨询.2007-2008年中国即时通信行业发展报告简版. http://www.iresearch.com.cn/html/Consulting/instant_messenger/Free_classid_16_id_1175.html

[2] 艾瑞市场咨询.2006年中国即时通信简版报告.http://www.iresearch.com.cn/Report/Free.asp?id=985

[3] EIM：针对企业级用户开辟即时通信市场.http://www.chinabyte.com/e/438/2012438.shtml

[4] 邹大斌.企业级IM应用悄然起步.计算机世界·技术与应用. 2009,06

[5] 网易学院.即时通信领域必将最终由XMPP协议一统天下.http://tech.163.com/07/0627/16/3I0Q86FG000917GT.html

[6] 张文茂,章淼,毕军,覃征.互联网即时消息(Instant Messaging, IM)的研究现状与展望.小型微型计算机系统.2005,20(5):56~61

[7] 腾讯RTX. http://rtx.tencent.com/rtx/index.shtml

[8] IBM Lotus sametime. http://www.ibm.com/software/lotus/sametime

[9] LiveCommunicationsServer. http://www.microsoft.com/china/office/livecomm/prodinfo/default.mspx

[10] IETF. Extensible Messaging and Presence Protocol (XMPP):Core. RFC3920.2004,10

[11] XMPP and Jabber. http://xmpp.org/about/jabber.shtml

[12] IETF. Mapping the Extensible Messaging and Presence Protocol (XMPP) to Common Presence and Instant Messaging(CPIM).RFC3922.2004,10

[13] IETF. End to End Signing and Object Encryption for the Extensible Messaging and Presence Protocol(XMPP).RFC3923.2004,10

[14] History of XMPP. http://xmpp.org/about/history.shtml

[15] IETF.A Model for presence and Instant Messageing.RFC2778.2000

[16] IETF. Instant Messaging/Presence Protocol Requirements.RFC2779.2000

[17] 苗凯.XMPP的安全机制分析.通信技术.2003.8:101~105

[18] 张云川.标准化的即时通信协议—SIMPLE和XMPP的对比研究.武汉科技大学学报.2005,28(4):375~377

[19] IEFT.SIP: Session Initiation Protocol.RFC3621

[20] IEFT. Session Initiation Protocol (SIP) Extension for Instant Messaging.RFC3428

[21] Cathleen Moore. XMPP vs SIMPLE: The race for messaging standards. May 23.2003. http://www.infoworld.com/article/03/05/23/21FExmpp_1.html

[22] F Osterfeld, M Kiesel, S Schwarz. A Semantic Archive for XMPP Instant Messaging. Semantic Desktop Workshop at the ISWC,2005

[23] 杨斌.XMPP协议分析与应用探讨.微型机与应用.2005,24(8):32~34

[24] IETF. Extensible Messaging and Presence Protocol(XMPP):Instant Messaging and Presence.RFC3921.2004,10

[25] XEP-0004.Data Forms.http://xmpp.org/extensions/xep-0004.html

[26] Jabber Software. Why your Business Should Use Enterprise Instant Messaging Now. http://www.instantmessagingplanet.como/public/article.pho/3491996. 2006,9

[27] 李明达.即时通信需要解决两大难题.通信世界.2007,11(28)

[28] 侯可.一种基于XMPP的企业即时消息技术.科技情报开发与经济.2008,18(26): 146~147

[29] XEP-0049.Private XML Storage.http://xmpp.org/extensions/xep-0048.html

[30] XEP-0083.Nested Roster Groups.http://xmpp.org/extensions/xep-0083.html

[31] XEP-0045.Multi-User Chat.http://xmpp.org/extensions/xep-0045.html

[32] XEP-0096.SI File Transfer.http://xmpp.org/extensions/xep-0096.html

[33] XEP-0047.In-Band Bytestreams.http://xmpp.org/extensions/xep-0047.html

[34] XEP-0050.Ad-Hoc Commands.http://xmpp.org/extensions/xep-0050.html

[35] Openfire.http://www.igniterealtime.org/projects/openfire/index.jsp

[36] Openfire开发文档.http://www.blogjava.net/yi88han/archive/2009/02/11/254203.html

[37] 潘凤,王华军,苗放,李刚.基于XMPP协议和Openfire的即时通信系统的开发.计算机时代.2008,3:15~19

[38] Gloox.http://camaya.net/

[39] Gloox技术分析.http://blog.csdn.net/qiuhong101/category/432064.aspx

[40] 张彦,夏清国.Jabber/XMPP技术的研究与应用.科学技术与工程.2007,7(6) :1032 ~1035

[41] 黄勇,万琴,黄晓萍.基于XMPP标准的即时消息系统及其应用.江西科学.2005,23 (6):776~780

[42] 樊燕红,谭香.基于XMPP的即时通信网关应用研究.电子技术用.2007,33(10):123~144

[43] Gloox源码分析.http://www.cppblog.com/woomsg/archive/2008/10/17/64260.aspx

[44] jive Software. Openfire Plugin Developer Guide. http://www.igniterealtime.org/builds/openfire/docs/latest/documentation/plugin-dev-guide.html

[45] 隋大鹏.Openfire源码分析.http://blog.csdn.net/racingtom/archive/2007/10/17/1829357.aspx

[46] Openfire插件开发.http://phoenixtoday.blogbus.com/logs/20285574.html

[47] Jive Software.Custom Database Integration Guide. http://www.igniterealtime.org/builds/openfire/docs/latest/documentation/db-integration-guide.html

[48] jive Softwareoft.Database Schema Guide. http://www.Igniterealtime.org/builds/openfire/docs/latest/documentation/database-guide.html

[49] XEP-0030.Service Discovery.http://xmpp.org/extensions/xep-0030.html

[50] XEP-0065.SOCKS5 Bytestreams.http://xmpp.org/extensions/xep-0065.html

[51] Extensible Messaging and Presence Protocol. http://en.wikipedia.org/wiki/Extensible_Messaging_and_Presence_Protocol

[52] 杜玲,苗放,李刚.XMPP协议研究及其在IM系统群组通信中的应用.湖南工程学院学报.2008,18(3)

攻读硕士学位期间发表的论文

[1] 付莎，耿峰.Windows平台下高性能可伸缩Winsock应用程序的研究.计算机应用研究(增刊).（已录用）

[2] 付莎.TCP穿越Cone型NAT的方法研究.电脑知识与技术.2009,5（12）：P3098-3100