东 北 石 油 大 学

课 程 设 计 报 告

课 程 软件工程课程设计

题 目 基于JS WebRTC API的P2P通话系统

院 系 计算机与信息技术学院软件工程系

专业班级 软工20-4

学生姓名 门鑫博、郑瑞蓬、盛钟锋

学生学号 200702940824、200702940816、200702940821

指导教师 王永安

2023年7月15日

**撰写格式要求【与“撰写格式式样”要求一致】**

一级标题黑体小二，段前段后2行；

二级标题黑体小三，段前段后1行；

三级标题黑体四号，段前1行；

四级标题黑体小四（非必需）；

正文是宋体小四，西文是Times New Roman ，1.2倍行距；

页眉按照本文样式；

目录单独用I，II等设置页号，正文用1,2,3….等设置页号，均为居中。

图标题是黑体五号，置于图下，图中文字是宋体五号（西文用Times New Roman）。表标题是黑体五号，置于表上，表中文字是宋体小五（西文用Times New Roman）。

本报告只给出模板参考，无需完全照搬局限，可酌情适当增删。

为规范报告样式，OO建模工具建议采用Visio或Rational Rose，框图、数据流图、程序流程图等建议采用Visio，进度甘特图建议采用Project或Visio，数据模型图可采用Visio或PowerDesigner。

每3-4人一组，每小组上交一个报告，报告页数应在60-90页。

东北石油大学课程设计任务书

课程 软件工程课程设计

题目 基于JS WebRTC的P2P通话系统

院系 计算机与信息技术学院 姓名 门鑫博、盛钟锋、郑瑞蓬

学号 200702940816 、 200702940821 、 200702940824

一、主要内容

包括需求分析

架构设计

功能实现

安全保障

测试与上线等

二、基本要求

在此系统中，应能实现：

文字聊天

语音通话

视频通话

数据流传输

三、主要参考资料

[1]刘映含. 基于WebRTC的实时视频图像增强技术研究[D].北京交通大学,2022:10.26-44.

[2]赵伯恺. 基于WebRTC的媒体服务器的设计与实现[D].北京邮电大学,2020:10.26-69.

[3]张灿. 基于WebRTC的实时图传系统的设计与实现[D].西安电子科技大学,2020:10.27-89.

[4]周华东.WebRTC实时通信开源技术介绍及实用案例[J].办公自动化,2020,25(04):14-16.

完 成 期 限： 18-20周

指 导 教 师： 王永安

专业负责人： 张强

2023年 07月15日

目 录

[第1章 可行性分析与项目开发计划报告 1](#_Toc140324689)

[1.1 项目概述 1](#_Toc140324690)

[1.2 可行性因素分析 3](#_Toc140324691)

[1.3 实施计划 7](#_Toc140324692)

[第2章 需求分析 13](#_Toc140324693)

[2.1 任务概述 13](#_Toc140324694)

[2.2 需求规定 16](#_Toc140324695)

[2.3 运行环境规定 23](#_Toc140324696)

[第3章 系统设计 25](#_Toc140324697)

[3.1 总体设计 25](#_Toc140324698)

[3.2 详细设计 33](#_Toc140324699)

[第4章 系统实现 47](#_Toc140324700)

[4.1 基础平台及资源 47](#_Toc140324701)

[4.2 主页面 49](#_Toc140324702)

[第5章 系统测试 64](#_Toc140324703)

[5.1 测试原则 64](#_Toc140324704)

[5.2 测试用例 65](#_Toc140324705)

[5.3 缺陷分析 69](#_Toc140324706)

[5.4 测试总结 70](#_Toc140324707)

[课程设计总结 71](#_Toc140324708)

[个人工作总结（门鑫博） 73](#_Toc140324709)

[个人工作总结（郑瑞蓬） 79](#_Toc140324710)

[个人工作总结（盛钟锋） 85](#_Toc140324711)

[参考文献 90](#_Toc140324712)

[附录1团队开发会议记录 7](#_Toc140324713)

[团队会议记录1 1](#_Toc140324714)

[团队会议记录2 2](#_Toc140324715)

[团队会议记录3 3](#_Toc140324716)

# 第1章 可行性分析与项目开发计划报告

## 1.1 项目概述

### 1.1.1 工作内容

基于WebRTC的通话系统是一项复杂的技术工程，需要多个领域的专业知识的综合运用。在开发过程中，需要考虑到架构设计、软件开发、功能实现和安全保障等多方面的因素，才能够开发出高质量、高性能、高安全性的通话系统。其中需要进行以下各种工作任务。

可行性分析：可行性分析的作用是评估新项目、新技术或新业务的可行性，确认项目是否具有实施的必要性和可行性，从而为后续项目开发工作提供基础。通过进行可行性分析，可以减少失败的风险，提高项目的成功率，为企业的发展提供保障。

需求分析：需求分析是整个开发过程中最为关键的步骤之一。对于基于WebRTC的通话系统而言，需求分析包括用户需求和系统需求两部分。通过调研用户需求，了解用户对语音和视频通讯服务的具体需求和期望，并根据需求制定相应的功能和接口。同时，还需要对系统需求进行分析，包括性能、稳定性、安全性等方面的要求。

架构设计：架构设计是基于WebRTC的通话系统开发中的另一个重要步骤。在进行架构设计时，需要考虑到系统的可扩展性、可靠性、易用性等因素。需要确定系统架构和技术方案，选择适当的服务器和数据库等组件，以确保系统能够满足用户需求并提供优良的使用体验。

软件开发：在进行软件开发时，需要根据需求分析和架构设计确定具体开发方案。软件开发包括前端和后端两部分。前端主要负责界面设计、用户交互等工作，需要选择合适的前端框架和技术实现。后端主要负责业务逻辑处理、数据存储等工作，需要选择合适的后端框架和数据库实现。

功能实现：基于WebRTC的通话系统需要实现多个核心功能，包括音视频传输、即时消息、静音和挂断、账户管理等。需要根据需求分析中所列出的功能要求，分别进行实现。在实现功能时，需要遵循统一的API接口规范，并考虑到系统的可扩展性和可维护性。

安全保障：为了确保基于WebRTC的通话系统的安全性和稳定性，需要进行多种安全保障措施。这些措施包括数据加密、用户认证、防火墙、DDoS攻击防御、备份和恢复等。需要评估系统安全性能和安全机制，并在开发过程中考虑到安全问题。

软件测试：在完成开发后，需要对整个系统进行测试，包括单元测试、功能测试、集成测试等。需要保证系统能够满足用户需求并提供良好的使用体验。在测试通过后，需要进行上线部署，将系统部署到生产环境中，并维护和优化系统。

总之，在基于WebRTC的通话系统开发中，需要进行多个步骤的工作，包括需求分析、架构设计、软件开发、功能实现、安全保障、测试与上线等。各项工作都需要专业人员参与和负责，才能开发出高质量、高性能、高安全性的通话系统，并为用户提供更好的使用体验。

### 1.1.2 主要参加人员

参与本项目的主要成员有郑瑞蓬、门鑫博、盛钟锋，其中由于小组成员过少，所以每个人都会分到很多职位，其中在JavaScript语言使用方面郑瑞蓬最熟练，而盛钟锋和门鑫博相对来说使用的不够熟练，在进行项目实现的过程中需要郑瑞蓬的指导和帮助。其中遇到不会的问题会进行小组讨论，最后由郑瑞蓬进行讲解。

### 1.1.3 产品

#### 1.1.3.1 程序

列出须移交给用户的程序的名称、所用的编程语言及存储程序的媒体形式，并通过引用相关文件，逐项说明其功能和能力。

表1-1 程序介绍

|  |  |
| --- | --- |
| 程序的名称 | 基于JS WebRTC API的P2P通话系统 |
| 所用的编程语言 | 前端：HTML CSS JS 后端：NodeJS |
| 存储程序的媒体形式 | 硬盘 |

#### 1.1.3.2 文件

在本次项目开发中共用到了如下的文件，且需要将文件解释说明给用户，文件介绍表格如下。

表1-2 文件介绍

|  |  |
| --- | --- |
| CallingState.js | 通话状态枚举 |
| MessagePackage.js | 通话信息封装 |
| MesssageType.js | 消息类型枚举 |
| ObserverCallBackType | 监视器事件类型枚举 |
| AppGlobal.js | 全局数据 |
| call-operation.js | 通话核心 |
| ConstantObserver.js | 监视器核心 |
| output-info-into-panel.js | 测试用控制面板 |
| WSConnection.js | WebRTCSocket功能封装 |
| 创建通道.js | WebRTC数据通道 |
| call-timer.js | 计时器模块 |
| callpage-UI.js | 通话页面UI |
| Object-prototype-toString2.js | 测试用扩展对象转字符串 |
| aside-controller-ui.js | 边栏UI数据计算 |
| friend-list.js | 好友列表管理 |
| send-message.js | 发送信息 |
| Message.ts | 信息封装 |
| MessageManager.ts | 信息管理类 |
| front-end-server.js | 前端发布页面服务器 |
| p2p-calling.html | 主页面 |
| SendMessage.js | 发送消息封装 |
| Mapper.ts | 通用映射类 |
| MessageOutput.js | 服务器日志 |
| PeerInfo.ts | 端信息封装类 |
| CallingManager.ts | 通话管理类 |
| PeerManager.ts | 端管理类 |
| TwoPeer.ts | 端对端封装 |

## 1.2 可行性因素分析

### 1.2.1 技术可行性

随着互联网技术的不断发展和普及，基于WebRTC开发的通话系统已经成为了一种新兴的通信方式。与传统的电话通讯相比，基于WebRTC开发的通话系统具有更加便捷、高效、低成本、可定制化等诸多优势，越来越受到人们的青睐。

浏览器平台：基于WebRTC技术的音视频通话需要使用支持WebRTC标准的浏览器，如Google Chrome、Mozilla Firefox等。通过浏览器提供的API接口实现音视频采集、编解码、传输和渲染等功能。

服务器平台：通话系统需要运行在服务器上，实现用户身份验证、消息转发、业务逻辑处理等功能。服务器平台需要具备高并发、低延迟、可扩展性等特点，常用的服务器平台有Java、Node.js、Python等语言和框架。

数据库平台：通话系统需要存储用户信息、通话记录、消息记录等数据，数据库平台需要支持高并发、事务管理、备份恢复等特点，常用的数据库平台有MySQL、MongoDB等。

网络平台：基于WebRTC技术的通话需要支持点对点连接和STUN/TURN/ICE协议等网络技术，实现NAT穿透、防火墙穿越等功能。

使用的编程语言：

前端语言：通话系统前端需要使用HTML、CSS、JavaScript等技术实现页面布局、样式设计和交互效果。同时，前端还需要使用框架来简化代码编写并提高开发效率。

后端语言：通话系统后端需要使用Node.js等语言和框架来处理用户请求、存储数据和实现业务逻辑。同时，后端还需要使

WebRTC相关语言：基于WebRTC技术开发音视频通话系统需要使用JavaScript和WebRTC API来实现音视频采集、编解码、传输和渲染等功能。

基于WebRTC的通话系统需要依赖于网络技术、编程语言、协议和相关软件开发工具等基础技术。例如，WebRTC技术可以帮助实现浏览器内的实时通信功能；HTML5技术可以使得网页具有更强的功能性和交互性；JavaScript语言可以帮助实现动态效果和用户交互等。同时，还需要考虑到服务器端的技术支持。

1、通话质量：基于WebRTC的通话系统在实际使用中需要保证通话质量。这涉及到网络传输速度、延迟、抖动、丢包率等因素的控制。为了提高通话质量，需要优化网络配置以及采用合适的编解码算法等措施，确保语音、视频的传输质量。

2、平台兼容：基于WebRTC的通话系统需要在多平台上运行，如Windows、Mac OS、Linux等不同的操作系统，以及各种类型浏览器，如Chrome、Firefox、Safari等。因此，在开发过程中需要考虑到平台兼容性问题，确保在不同的平台上都能够正常运行。

3、安全性：基于WebRTC的通话系统需要保证通话的安全性，防止他人窃听通话内容、恶意攻击等。为此，需要采用加密技术来保护通话内容的安全，并且确保通话系统本身没有安全漏洞，以避免被黑客攻击。

4、可扩展性：基于WebRTC的通话系统需要具备良好的可扩展性，以适应未来不断增长的用户需求。例如，可以通过添加新的功能模块或者优化现有模块的方式来保持其创新和竞争力。

总的来说，基于WebRTC的通话系统技术可行性分析需要考虑到基础技术、通话质量、平台兼容、安全性和可扩展性等多方面因素。只有在技术实现上充分考虑到这些因素之后，才能保证该系统的稳定性、安全性和可靠性，以满足不断变化的用户需求。

### 1.2.2 经济可行性

通过网络技术实现语音、视频通话等功能。对于该系统的技术可行性分析需要考虑到以下几点

1、开发成本：基于WebRTC的通话系统需要使用相关技术工具和编程语言，如WebRTC、JavaScript、HTML5等。开发团队需要具备相应的技术能力，并确保该系统可以运行在主流的浏览器上，如Chrome、Firefox、Safari等。因此，开发该系统需要投入大量的资金和人力资源，包括开发团队的薪资、设备购置费用、办公场地租金等。

2、运营成本：基于WebRTC的通话系统的运营成本主要包括服务器租赁费用、带宽费用、系统维护费用以及推广费用等。服务器租赁和带宽费用是运营成本中的重要部分，因为基于WebRTC的通话系统需要一个稳定、高效的服务器来保障通话质量。另外，系统的维护费用也需要考虑到，例如定期更新系统版本、修复漏洞等方面的开销。

3、预期收益：预期收益是评估该系统经济可行性的关键因素之一。根据该系统的商业模式，可能的预期收益方式包括用户付费、广告收入、增值服务等。例如，在用户付费方面，该系统可以提供基本的免费服务和高级的付费服务，以吸引用户购买高级服务。在广告收入方面，该系统可以通过展示相关广告来获取收益。在增值服务方面，该系统可以提供一些额外的服务，如云存储、文件传输等，以增加用户黏性和收益。

4、盈利模式：基于WebRTC的通话系统的盈利模式是评估其经济可行性的重要因素之一。例如，在盈利模式方面，对于一些创业公司来说，他们可能需要通过用户付费或广告收入等方式来获取收益。另外，还需要考虑到系统的运营成本、维护成本等因素，以确保在长期经营中仍能够实现盈利。

5、收益风险分析：收益风险分析是评估该系统经济可行性的重要因素之一。收益风险主要来自于市场需求变化、竞争压力等因素，因此需要在预期收益中考虑到不同情况下的收益变化。同时，也需要考虑到系统的运营成本、维护成本等因素，以避免出现收益不足以覆盖成本的问题。

6、经济方面考虑到初期开发人员为我们组内的三名成员，以及长期的维护和升级成本。我们计划从隐藏式广告和其他途径获得收入，同时也通过用户数量等方面进行不同的盈利方式来获得相应的费用收入。同时我们还需遵守相关的隐私和数据保护法规，在用户聊天时做到不泄露用户聊天隐私

总的来说，基于WebRTC的通话系统经济可行性分析需要考虑到开发成本、运营成本、预期收益、盈利模式和收益风险等因素。只有在充分评估这些因素之后，才能够评估该系统的经济可行性，并制定相应的商业模式和运营策略，以确保项目能够长期稳定发展并实现盈利。

### 1.2.3社会可行性

基于WebRTC的通话系统是现代人们日常沟通和交流必不可少的一种方式，具有很强的社会可行性。

1、满足社会需求：在当今社会中，人们对通讯工具的需求越来越高。随着科技的发展，基于WebRTC的通话系统成为了人们实现远距离交流的重要手段。例如，通过WebRTC电话、视频会议等功能，人们可以轻松地进行远程办公、在线教育、远程医疗等活动，提高了人们的工作效率和生活质量。

2、促进社交互动：基于WebRTC的通话系统也可以帮助人们进行社交互动，增进人与人之间的联系和沟通，扩大社交圈，丰富社交体验，适应了现代社会不断变化的社交需求。例如，社交媒体平台上的观看直播、发布信息等功能，可以帮助人们更好地与朋友、家人和同事保持联系。

3、便于信息传播：基于WebRTC的通话系统还可以帮助信息更快地传达到目标用户。例如，通过网络电话、视频会议、语音消息等形式，人们可以更加便捷地获取信息，并迅速和他人分享。这为各行业的信息传播和宣传带来了新的机遇，使得信息能够更加全面、准确地被传递出去。

4、提高工作效率：基于WebRTC的通话系统也可以帮助人们更快速地完成工作任务。例如，通过屏幕共享、远程协作、远程控制等功能，团队成员不受时空限制，可以更加高效地协同工作。这提高了工作效率，同时也降低了企业的运营成本。

5、促进数字化生活：随着数字化生活的普及，基于WebRTC的通话系统也为数字生活提供了重要的支持。例如，在线学习、在线购物、在线娱乐等领域，人们可以更加便捷地享受数字化生活带来的便利和乐趣。同时也为数字化生活的发展提供了有力的技术保障。

它满足了人们日常生活中对通讯工具的需求，增进了人们之间的沟通和联系，提高了工作效率和生活质量，并推动了数字化生活的普及。在未来，随着技术的不断创新和发展，基于WebRTC的通话系统还将继续拓展其应用领域和市场规模。

### 1.2.4法律可行性

随着其在日常生活中的不断普及和应用，它也面临了一系列法律问题。因此，在进行基于WebRTC的通话系统法律可行性分析

1、隐私保护：基于WebRTC的通话系统涉及到用户的个人信息、通话内容等敏感信息，因此需要对用户的隐私进行有效保护。为此，该系统需要遵守相关法律规定，确保用户的个人信息不被泄露或滥用。

2、通讯监控：在某些情况下，政府或执法机构可能需要对通讯内容进行监控或调查。针对这种情况，基于WebRTC的通话系统需要遵守国家相关法律法规的规定等，并提供必要的技术支持，以便政府和执法机构可以进行必要的监管和调查。

3、知识产权保护：在使用基于WebRTC的通话系统时，可能会涉及到知识产权问题。例如，用户在通话过程中使用了他人所拥有的专利技术或著作物等。为此，该系统需要遵守知识产权相关法律规定等，并确保用户在使用该系统时不会侵犯他人的知识产权。

基于WebRTC的通话系统的法律可行性分析需要考虑到个人隐私保护、知识产权保护等多个方面的问题。只有在系统开发和运营过程中充分遵守相关法律法规的规定，才能保证系统的合法性和稳定性，同时也能够保护用户的合法权益和利益。

### 1.2.5环保可行性

基于WebRTC的通话系统在实际应用中需要考虑到环境可行性，它涉及到该系统是否能够适应不同地区、不同文化和不同语言等多种环境因素。

1、地域因素：基于WebRTC的通话系统需要考虑到不同地区用户的网络环境、设备硬件标准、通信成本等因素。例如，在中国，由于互联网监管相对较严格，因此该系统在开发和运营过程中，需要遵守相关政策法规，并且针对中国的网络环境进行优化和改进。

2、语言因素：基于WebRTC的通话系统需要支持多种语言，以满足全球用户的需求。因此，在开发过程中需要考虑到国际化和本地化的问题，以确保用户可以使用他们所熟悉的语言进行通讯和交流。

3、文化因素：基于WebRTC的通话系统需要考虑到不同文化背景下人们的习惯、价值观念和审美等因素。例如，在某些国家和地区，人们更喜欢使用视频聊天，而在另一些地区则更倾向于使用语音通话。因此，在设计系统的交互界面、功能配置等方面，需要注重用户体验和文化适应性。

4、设备因素：基于WebRTC的通话系统需要考虑到不同设备的硬件标准和性能要求。例如，在手机和PC电脑上使用该系统时，需要考虑到屏幕大小、分辨率、音频设备和摄像头等因素，以确保通话质量和用户体验。

5、网络因素：基于WebRTC的通话系统需要考虑到不同地区网络环境的差异。例如，在某些地区，网络带宽和稳定性相对较差，因此需要针对这些情况进行优化和调整，以确保通信质量和可靠性。

基于WebRTC的通话系统环境可行性分析需要充分考虑到地域因素、语言因素、文化因素、设备因素和网络因素等多个方面的问题。只有在系统开发和运营过程中充分考虑到这些因素，并且针对不同的环境因素进行优化和改进，才能保证该系统的稳定性和可靠性，同时也能够满足全球用户的需求。

## 1.3 实施计划

### 1.3.1 工作任务的分解与人员分工

基于WebRTC的通话系统开发是一项复杂的技术工程，需要多个领域的专业知识的综合运用。在开展开发工作时，需要对任务进行分解，并确定合适的人员分工，以确保项目能够顺利地实施和落地。下面是基于WebRTC的通话系统开发工作任务的分解：

1、需求分析阶段：

在需求分析阶段，首先要根据市场需求，进行用户调研和数据分析，了解用户的需求和期望，制定项目需求规格说明书，并进行技术可行性分析、经济可行性分析、法律可行性分析等。具体任务如下：

* 市场调研：了解市场需求和竞争情况。
* 用户调研：跟踪用户需求，收集用户反馈和建议。
* 数据分析：对市场数据进行分析，预测市场趋势。
* 需求分析：根据用户需求制定项目需求规格说明书。
* 可行性分析：进行技术可行性分析、经济可行性分析、法律可行性分析等。

2、设计和开发阶段：

在设计和开发阶段，需要进行系统设计和编码实现，包括前端UI设计、后端程序开发、安全性保障等方面的工作。具体任务如下：

* 系统设计：进行系统架构设计和技术选型。
* 前端UI设计：设计用户界面，制作交互效果。
* 后端程序开发：负责业务逻辑处理、数据存储等工作，实现各项功能。
* 自动化测试：对软件进行自动化测试，提高软件质量和稳定性。

3、测试和上线阶段：

在测试和上线阶段，需要对整个系统进行测试和维护，并将系统部署到生产环境中，进行客户端集成、发布、更新等工作。具体任务如下：

* 单元测试：对代码进行单元测试，保证功能完整性。
* 集成测试：进行前后端集成测试，确保系统正常运行。
* 发布部署：将系统部署到生产环境中，并进行客户端集成、发布、更新等工作。
* 维护升级：对系统进行维护和升级，修复漏洞和缺陷。

4、项目管理阶段

在项目管理阶段，需要进行项目计划、资源调度、团队合作、风险评估等工作，保证项目能够按时完成并达到预期目标。具体任务如下：

* 项目计划：制定项目计划，确定里程碑和时间节点。
* 资源调度：对人力、物力进行资源调度，保证项目实施的顺利进行。
* 团队合作：加强团队协作，提高工作效率和质量。
* 风险评估：对可能出现的风险进行评估和规避。

综上所述，在基于WebRTC的通话系统开发过程中，需要根据任务的不同特点和要求，对任务进行分解。通过合理的任务分解和人员分工，可以确保每个人都有明确的职责和任务，并且整个开发过程能够有序进行，从而提高工作效率和质量，确保项目顺利完成。

|  |  |
| --- | --- |
| 成员 | 角色 |
| 门鑫博 | 项目经理、程序员、前端工程师、后端工程师、测试员、发布工程师 |
| 郑瑞蓬 | 设计师、程序员、测试员、架构师、前端工程师、后端工程师、运维工程师 |
| 盛钟锋 | 业务人员、前端工程师、后端工程师、程序员、测试员、客户端集成工程师 |

表1-3 任务分配

### 1.3.2 进度

甘特图是一种项目管理工具，用于可视化项目的进度和计划。它通过横轴表示时间，纵轴表示项目任务，以条形图的形式展示每个任务的开始时间、结束时间和持续时间，同时可以显示任务之间的依赖关系。以下为本项目的进度甘特图。



图1-1 项目进度甘特图

### 1.3.3 成本效益评估

其中该项目共三名成员，用时为三周，根据市场行情，每人每天获得1500到2000元的劳务费，资料费共计5000元，设备费用主要用在为成员购买电脑，共计3万元，服务器租用金额为随用户数量而进行变化。

首先需要计算项目的总投资额，包括硬件和软件设备、人员费用、市场推广费用等方面的成本。大致需要项目的总投资额为200万元。

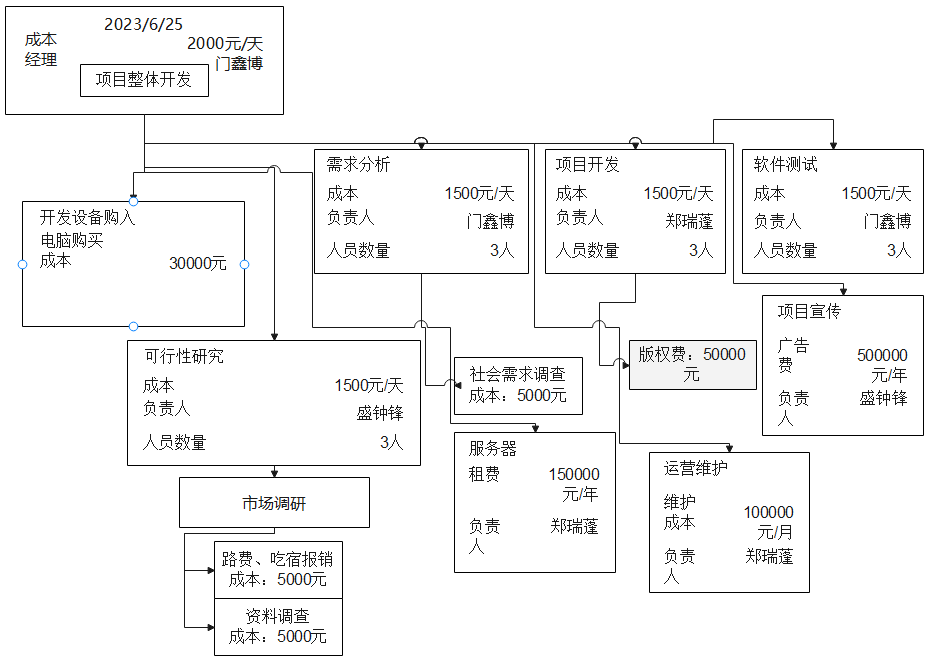


图1-2 成本估算图

### 1.3.4 关键问题

能够影响整个基于WebRTC的通话系统项目成败的关键问题、技术难点和风险包括以下几个方面：

竞争压力和市场需求：竞争对手众多，市场需求不明确或需求变化快速等问题，会导致用户选择其他应用而非本项目，从而影响项目的推广和盈利。

技术难点和可靠性问题：音视频传输质量受网络带宽限制，通话过程中可能出现卡顿、断线等问题，同时需要考虑安全性等技术难点对项目的影响。

用户数据安全和隐私保护：用户的个人信息和隐私需要得到有效的保护，避免出现泄露等情况，否则会严重影响用户体验和信任度。

项目可行性和投入产出比：在进入实施阶段前，需要进行充分的市场调研、技术评估、经济效益分析等工作，以提高项目的可行性，避免因为错误的判断而造成投资和资源浪费。

这些问题对项目的影响主要体现在以下几个方面：

影响项目的推广和盈利能力，可能导致项目不能回收足够的投资成本，甚至亏损。

影响项目的可靠性和用户体验，使得用户对该应用失去信心，导致用户流失率增加。

存在数据安全问题，可能导致用户信息泄露和商业秘密外泄，并对项目形象和信誉造成负面影响，严重情况下可能会涉及法律诉讼。

对项目管理和资源调度产生影响，导致项目推迟或停滞不前，浪费人力和物力资源，最终影响项目的成功与否。

因此，在项目实施过程中需要充分考虑这些问题的影响，采取相应的措施来防范风险、提高项目的成功率。

关键问题：

影响整个基于WebRTC的通话系统项目成败的关键问题包括以下几个方面：

项目可行性和市场需求：在进入实施阶段前，需要进行充分的市场调研、技术评估、经济效益分析等工作，以提高项目的可行性，并适应市场需求变化。

竞争压力和用户体验：市场上已经存在一些类似的通话系统产品，因此项目需要考虑如何与竞争对手进行竞争，并吸引更多的用户使用本产品。同时，需要提供优秀的用户体验，从而增加用户黏性和忠诚度。

安全性和数据隐私：基于WebRTC的通话系统需要涉及到用户的个人信息和隐私，因此需要确保系统的安全性，防止黑客攻击和数据泄露等安全问题。

营收模式和盈利能力：基于WebRTC的通话系统需要考虑如何盈利，例如用户付费、广告收入等方式。需要分析用户需求和市场趋势，并制定相应的营销策略，以提高盈利能力。

团队建设和管理：项目团队需要具备专业知识和经验，能够有效地协作和沟通。同时，需要充分考虑人员流动性和管理问题，以确保项目的顺利推进。

以上关键问题会影响基于WebRTC的通话系统项目的开发和部署，如果解决不好这些问题，可能会导致项目无法正常运行，影响用户体验并最终影响项目的成功与否。因此，在项目实施过程中需要充分考虑这些关键问题，并采取相应的措施，例如加强市场调研和需求分析、提升产品质量和用户体验、加强安全保护、制定科学合理的营销策略等，以提高项目的成功率和盈利能力。

风险：

影响整个基于WebRTC的通话系统项目成败的风险包括以下几个方面：

技术风险：基于WebRTC的通话系统需要涉及到音视频传输、网络带宽等技术，因此可能存在技术难点和问题，如数据传输速度、网络质量、协议兼容性等。这些风险可能会导致系统无法正常运行，从而给用户带来不良的体验。

竞争风险：市场上已经存在一些类似的通话系统产品，项目需要解决如何与竞争对手进行竞争，并吸引更多的用户使用本产品。如果不能有效竞争，可能会导致用户流失，最终影响项目的盈利能力。

安全风险：基于WebRTC的通话系统需要涉及到用户的个人信息和隐私，因此需要确保系统的安全性，防止黑客攻击和数据泄露等安全问题。如果出现安全漏洞，可能会严重影响用户体验和信任度。

人员风险：项目需要依赖专业人员进行开发、测试、运维等工作，而人员流动性较高，团队建设和管理存在风险。如果人员流失或者新员工能力不足，可能会影响项目的进展和质量。

经济风险：项目需要投入大量的资金和资源才能进行研发、测试和推广，如果市场需求不足或预算计划出现偏差，可能会导致投资损失和回报不佳。

因此，在基于WebRTC的通话系统项目中，需要充分考虑这些风险，并采取相应的措施来规避或减少风险的影响。例如，在技术方面，可以对系统进行充分的测试和评估，确保系统可靠性和稳定性；在竞争方面，可以通过营销策略和用户体验等方面提高产品的吸引力；在安全方面，可以采用专业的安全技术和措施，确保用户信息安全；在人员方面，可以加强团队建设和管理，提高人员素质和稳定性；在经济方面，可以制定科学合理的预算计划，降低经济风险的影响。

技术难点：

影响整个基于WebRTC的通话系统项目成败的技术难点包括以下几个方面：

音视频传输：基于WebRTC的通话系统需要涉及到音视频传输，需要保证音视频质量和实时性。同时，网络带宽和延迟等因素也会影响音视频传输效果。

网络通信协议：基于WebRTC开发的通话系统需要兼容多种网络通信协议，如HTTP、TCP、UDP等，这可能会增加项目的复杂度和难度。

浏览器兼容性：不同的浏览器支持的技术和版本不同，需要在多种浏览器上进行充分测试，确保系统能够兼容各种浏览器，并且在不同的设备上表现一致。

安全性：基于WebRTC的通话系统存在安全问题，比如黑客攻击、信息泄露等。因此，系统需要采取一系列措施保护用户数据的安全。

用户体验：通话系统需要提供优秀的用户体验，包括清晰的音视频、快速的连接、稳定的通话等。如果用户体验不佳，可能会导致用户流失和口碑下降。

以上技术难点会影响基于WebRTC的通话系统项目的开发和部署，如果解决不好这些难点，可能会导致项目无法正常运行，影响用户体验并最终影响项目的成功与否。因此，在项目实施过程中需要充分考虑这些技术难点，并采取相应的措施，例如使用高效的音视频编码和传输算法、采用流媒体技术、加强安全保护等，以提高系统的性能和稳定性，确保用户体验和使用效果。

# 第2章 需求分析

## 2.1 任务概述

### 2.1.1 课题意义

基于WebRTC的P2P视频通话系统在社会、安全和文化等方面都具有重要意义和目的。从社会角度看，该系统可以促进人们之间的跨地域、跨国界交流和合作，为消费者、教育、医疗、政府等领域提供更加便捷、高效的服务。比如，利用该系统，医生可以通过视频通话远程诊断患者病情，解决医疗资源不足的问题；教师可以通过视频直播授课，面向全球范围内的学生进行教学；政府官员可以通过视频会议协商解决区域性问题，提高行政效率。这些应用场景都可以通过基于WebRTC的P2P视频通话系统得到优化和改善。

从安全角度看，该系统采用了端对端加密技术，保证了通信内容的隐私性和安全性。在传统的中心服务器转发架构中，数据需要经过服务器中转，存在被截获和窃取的风险，而基于WebRTC的P2P技术则是点对点直接连接，数据只有在双方之间传输，中间不存在第三方，则大大降低了数据泄露的风险，保护了用户的隐私。此外，该系统还采用了防火墙穿透技术，可以通过网络屏障进行数据传输，提高了系统的抗干扰和稳定性。

从文化角度看，该系统可以促进不同国家、不同地区，甚至不同语言、文化之间的交流和理解，增强人们的文化认知和多元化视野，有助于缩小文化差距，推动世界文明的交融和发展。

在技术背景方面，该系统采用了当前最先进的WebRTC技术，并结合了js和nodejs等语言特点，实现了前后端分离、跨平台和易扩展等优点，具有良好的用户体验和开发可维护性。此外，该系统还支持流媒体服务、视频编码和解码、自适应比特率控制等技术，提升了视频通话的质量和稳定性。

综上所述，这种新型的视频通话方式可以为人们提供更加便捷、高效、安全的服务和交流方式，同时也具有很大的商业潜力和市场前景，是一项值得深入研究和推广的技术。

### 2.1.2 用户的特点

本软件是一款基于WebRTC的P2P视频通话系统，其最终用户主要包括以下几个特点：

业务需求：该软件适用于需要进行远程视频通话、在线教育和娱乐活动等场景。因此，最终用户主要来自于医疗、教育、企业和消费者等领域，对视频通话的质量和稳定性有较高要求。

技术水平：由于是一种新型的通信技术，因此用户群体通常具备一定的计算机技术和网络知识。在医疗、教育、企业等领域使用的用户相对来说更专业一些，他们通常会对电脑和网络设备有一定的了解和掌握。

年龄层次：年龄方面，最终用户也存在着差异。在医疗、教育、企业等领域使用的用户通常比较成熟，年龄段范围广泛；而在个人娱乐、社交等领域使用的用户则普遍年轻化。

使用频率：软件的使用频率与用户的需求及使用场景密切相关。在医疗和教育领域，视频通话的需求相对较高，预计使用频率也较大。而在个人娱乐和社交方面的应用则较为分散，使用频率相对较低。

针对这些最终用户特点，我们需要培训操作人员和维护人员以保证软件的操作和维护质量：

操作人员：操作人员需要具备基本的计算机使用技能，如使用麦克风、摄像头等外围设备、掌握基础网络知识、会使用基础的软件功能。另外，由于该软件主要是远程视频通话，因此操作人员还需要掌握一定的沟通技巧和语言表达能力，以提供更好的服务。

维护人员：维护人员需要具备计算机网络、服务器管理、数据库等相关专业知识和技能，能够对软件系统进行故障排除、性能优化等工作。此外，还需要了解数据加密、身份认证、防火墙穿透等安全技术，并能够有效地应对网络威胁和黑客攻击等安全问题。

教育水平：操作人员和维护人员的教育水平也需要有所考虑。由于该软件涉及到医疗、教育等领域，因此需要考虑相关行业的专业知识和教育背景。同时，操作人员和维护人员还需要接受定期的培训和技能提升，以保持工作能力和适应新技术。

最后，根据预计的用户特点和使用频率，我们也需要对软件进行相应的设计和开发，以满足用户的需求和提高软件的竞争力。

### 2.1.3 假定和约束

进行本软件开发工作的假定和约束有很多，以下列举了几个主要的：

经费限制：本软件开发需要一定的经费支持。因为本软件是基于WebRTC的 P2P 视频通话系统，需要使用到比较先进的技术，并且需要进行大量的测试、维护和升级等工作。因此，需要预留足够的经费用于软件开发、测试、发布和维护。

开发期限：在进行本软件开发前，需要有一个明确的开发周期安排。开发周期包括需求分析、设计、编码、测试等不同阶段，每个阶段的时间都需要合理地安排。这样可以避免项目延期，保证软件的及时上线。

技术约束：本软件采用的是WebRTC技术，在开发过程中需要考虑不同浏览器、平台之间的兼容性问题。同时，由于涉及视频通话等功能，还需要考虑网络带宽、视频质量、延迟等方面的问题。开发人员需要具备相关的技术专长，能够解决这些技术难题。

用户需求约束：软件的开发需要根据最终用户的具体需求来进行设计和开发。因此，需要进行一定的市场调查和用户需求分析，以确保软件的功能和性能能够满足用户的要求。

安全约束：由于本软件涉及到用户的个人信息和隐私，因此需要采取一定的安全措施来保护用户的数据和隐私。开发人员需要具备相关的安全技术知识，能够设计出安全可靠的软件系统。

合规约束：在软件开发过程中，需要遵守相关法律法规、行业标准和隐私协议等方面的约束。例如，需要遵守《网络安全法》等法律法规，严格保护用户数据的隐私和安全。

人力资源约束：软件开发需要有一支专业的开发团队来完成。这些人员需要具备不同的专业技能和工作经验，包括项目管理、前端开发、后端开发、数据库管理、测试等方面的技能。因此，在软件开发过程中，需要考虑到人力资源的配置问题，合理分配开发任务和工作量。

综上所述，进行本软件开发工作的假定和约束非常多，开发人员需要充分了解这些约束条件，并且进行合理规划和设计，以确保软件的开发顺利进行。此外，需要对不同约束条件进行风险评估和管理，以应对可能出现的挑战和问题。

### 2.1.4 参考资料

[1]刘映含. 基于WebRTC的实时视频图像增强技术研究[D].北京交通大学,2022,10.26-44.

[2]赵伯恺. 基于WebRTC的媒体服务器的设计与实现[D].北京邮电大学,2020,10.26-69.

[3]张灿. 基于WebRTC的实时图传系统的设计与实现[D].西安电子科技大学,2020,10.27-89.

[4]周华东.WebRTC实时通信开源技术介绍及实用案例[J].办公自动化,2020,25(04):14-16.

[5]袁永胜,赵魁元,肖峥瑜等.基于ICE框架的P2P技术在通讯领域的研究[J].价值工程,2022,41(23):145-147.

[6]鲍薇.P2P技术的分析与研究[J].电脑迷,2020(07):5.

[7]许至晶.JavaScript在WebRTC开发的应用研究[J].信息记录材料,2022,23(12):242-244+248.

[8]向桂玲.JavaScript技术在WebRTC网页中的应用研究[J].信息记录材料,2022,23(04):145-147.

[9]张博,于海洋.服务器端JavaScript技术分析[J].信息与电脑(理论版),2021(04):19-20.

[10]宋文瑞.基于JavaScript技术实现的面向对象编程方法[J].数码世界,2021(02):9-11.

## 2.2 需求规定

### 2.2.1 对功能的规定

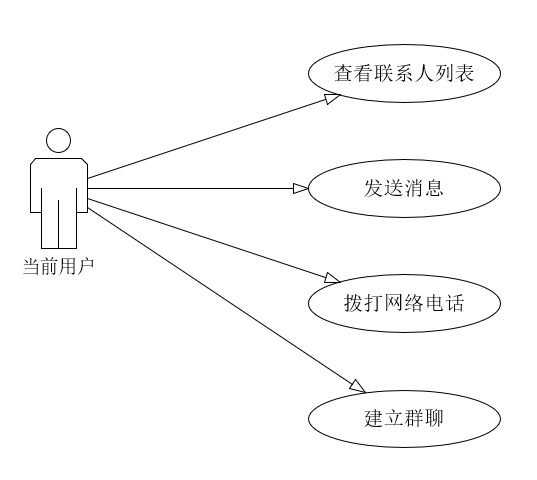


图2-1 系统用例图

表2-1 系统用例描述表

|  |  |
| --- | --- |
| 和其他用户互交 | Use Case：和其它用户互交  范围：视频通话系统  主要参与者：系统用户  前置条件：登录系统  后置条件：和其他用户进行文字或通话交互  基本事件流：  1.用户鼠标点击好友消息框  2.弹出好友聊天页面  3.输入光标移动到输入框内  4.与用户进行交互  5.用例终结  发生频率：经常 |
| 备选事件流：  1.与用户交互前可以点击退出按钮退出页面  2.可以关闭客户端浏览器  发生频率：偶尔 |
| 联系人 | Use Case：联系人  范围：视频通话系统  主要参与者：系统用户  前置条件：与用户交互  后置条件：对联系人进行管理  基本事件流：  1.用户进行联系人管理  2.弹出联系人管理页面  3.可以进行联系人查询、添加、修改、删除等操作  4.点击确定按钮  5.用例终结  发生频率：经常 |
| 备选事件流：  1.点击确定按钮前可以点击返回主页面  2.可以关闭客户端浏览器  3.发生错误时提示错误信息并返回主页面  发生频率：偶尔 |
| 好友列表 | Use Case：好友列表  范围：视频通话系统  主要参与者：系统用户  前置条件：与用户交互  后置条件：查看好友列表、选择对等端  基本事件流：  1.用户登录系统  2.展示好友列表  3.好友列表自动更新  4.用例终结  发生频率：经常 |
| 备选事件流：  1.可以关闭客户端浏览器  2.发生错误时提示错误信息并返回主页面  发生频率：偶尔 |
| 发送消息 | Use Case：发送消息  范围：视频通话系统  主要参与者：系统用户  前置条件：与用户交互  后置条件：查看好友列表、选择对等端  基本事件流：  1.用户点击聊天对等端  2.展示交互框  3.点击输入框  4.输入将要发送的消息  5.点击发送按钮  6.用例终结  发生频率：经常 |
| 备选事件流：  1.可以关闭客户端浏览器  2.发生错误时提示错误信息并返回主页面  3.点击发送按钮前点击退出将不会发送消息  发生频率：偶尔 |
| 打视频电话 | Use Case：打视频电话  范围：视频通话系统  主要参与者：系统用户  前置条件：与用户交互  后置条件：利用数据通道传递信息、接通、拒接、挂断  基本事件流：  1用户点击视频通话  2.展示视频通话界面  3.点击挂断可挂断电话  4.媒体数据开始由stun-turn服务器传输到对等端  5.用例终结  发生频率：经常 |
| 备选事件流：  1.发生错误时提示错误信息并返回主页面  2.可以关闭客户端浏览器  3.发生频率：偶尔 |
| 群聊 | Use Case：群聊  范围：视频通话系统  主要参与者：系统用户  前置条件：与用户交互  后置条件：查询、修改、添加、删除群聊  基本事件流：  1.进入群聊管理  2.弹出群聊管理页面  发生频率：经常 |
| 备选事件流：  1.发生错误时提示错误信息并返回主页面  2.可以关闭客户端浏览器  发生频率：偶尔 |
| 查询联系人 | Use Case：查询联系人  范围：视频通话系统  主要参与者：系统用户  前置条件：联系人  后置条件：无  基本事件流：  1.在联系人管理页面点击查询联系人  2.点击输入框  3.输入要查询的联系人昵称  4.点击查询按钮  5.用例终结  发生频率：经常 |
| 备选事件流：  1.发生错误时提示错误信息并返回主页面  2.可以关闭客户端浏览器  3.点击查询按钮前点击退出系统将不会进行查询操作  发生频率：偶尔 |
| 修改联系人 | Use Case：修改联系人  范围：视频通话系统  主要参与者：系统用户  前置条件：联系人  后置条件：无  基本事件流：  1.在联系人管理页面点击修改联系人  2.修改被选择的联系人信息  3.点击确定  4.联系人信息修改完毕  5.用例终结  发生频率：经常 |
| 备选事件流：  1.发生错误时提示错误信息并返回主页面  2.可以关闭客户端浏览器  3.点击确定按钮前退出界面将不对联系人进行修改操作  发生频率：偶尔 |
| 接通电话 | Use Case：接通电话  范围：视频通话系统  主要参与者：系统用户  前置条件：打视频电话  后置条件：无  基本事件流：  1.在收到通话后点击接通  2.用例终结  发生频率：经常 |
| 备选事件流：  1.发生错误时提示错误信息并返回主页面  2.可以关闭客户端浏览器  发生频率：偶尔 |
| 挂断电话 | Use Case：挂断电话  范围：视频通话系统  主要参与者：系统用户  前置条件：打视频电话  后置条件：无  基本事件流：  1.被呼端在接收到通话后点击挂断  2.呼叫端点击挂断  3.用例终结  发生频率：经常 |
| 备选事件流：  1.发生错误时提示错误信息并返回主页面  2.可以关闭客户端浏览器  发生频率：偶尔 |

### 2.2.2 响应速度

该系统需要保证高速响应，即在用户发出请求后能够及时返回结果,一下是本系统对响应速度的规定：

最大响应时间：2s

平均响应时间：经过测试为0.5s,用于评估系统的性能

响应时间分布：经过测试消息的发送与接收99%的响应时间不超过1s

通过规定以上响应时间，将有助于提高用户体验和满意度。

### 2.2.3 稳定性

该系统需要具有很高的稳定性，以保证用户可以在不同环境下进行视频通话，并且不会因为网络波动而影响通话质量，本系统对稳定性的规定如下：

可用性：经过测试系统的可用性为99%，即系统每年最多允许有1%的不可用时间；

故障率：经过测试系统的故障率低于0.1%；

恢复时间：保障系统在发生故障后30s内恢复运行。

### 2.2.4 拓展性

该系统需要具有一定的拓展性，以适应未来的扩展需求。例如支持更多的用户同时在线、支持更多的语言、支持更多的设备和平台等，本系统对拓展性的规定如下：

扩展能力：经过测试系统最大支持同时在线用户数为1000人；

响应时间增长率：经过测试系统在负载翻倍时，响应时间增长不超过50%；

扩展资源成本效益：分析系统扩展所需的硬件、人力和维护成本，并确保扩展是经济合理的。

### 2.2.5 可维护性

该系统需要具有较高的可维护性，即易于管理、维护和更新。在系统出现问题时，能够及时进行修复和升级，本系统对可维护性的规定如下：

代码可读性：使用代码静态分析工具检查代码规范性，并设定达到一定代码可读性评分的要求

故障处理时间：保障系统在接收到故障报告后在24小时内给予初步响应，并在48小时内完成修复

变更管理：要求每个变更都需要进行评估和测试，变更引入问题的比例不超过5%

### 2.2.6 兼容性

该系统需要兼容不同的操作系统和浏览器，以满足不同用户的使用需求，本系统对兼容性的规定如下：

平台兼容性：经过测试系统在 Windows、Mac 和 Linux 操作系统上都能够完全兼容

浏览器兼容性：设定目标支持的浏览器版本范围，并确保系统在这些版本的浏览器上能够正常运行

移动设备兼容性：手机、电脑、平板电脑等

数据格式兼容性：导入和导出数据时按照json、XML格式，确保系统能够正确解析和处理这些格式的数据

这将有助于确保系统具有更广泛的适用性和灵活性。

### 2.2.7 总结

WebRTC的P2P视频通话系统需要满足多种非功能性需求，包括响应速度、稳定性、拓展性、可维护性、兼容性等方面的需求。这些需求对于系统的使用和管理都有重要影响，因此在开发系统时需要注重这些方面的设计和实现，以确保系统的高效运行和用户体验。

### 2.2.8 输入输出要求

#### 2.2.8.1发送消息输入输出：

输入要求：

目标用户/群组：需要指定要发送消息的目标用户或者群组。

消息内容：需要提供要发送的消息内容。

发送者身份验证：发送方需要登录后才可发送消息

消息的优先级：根据发送方与接收方的关系对发送消息的优先级进行计算，优先级高的消息优先发出。

输出要求：

发送状态：返回消息是否成功发送的状态，可以是成功或者失败。

错误信息：如果发送状态为失败，可以返回相应的错误信息，方便定位问题。

消息ID：返回发送消息的消息ID，方便对消息进行快速定位，查看消息历史。

时间戳：返回消息的具体发送时间，监控系统的性能。

#### 2.2.8.2视频通话输入输出要求：

输入要求：

主叫方和被叫方信息：需要提供主叫方和被叫方的相关信息，如用户ID、用户名等。

通话发起时间：记录通话的发起时间，用于计算通话时长等。

视频参数：包括视频编解码器、分辨率、帧率等参数，用于设置视频通话的画质和性能。

音频参数：包括音频编解码器、采样率、音频通道数等参数，用于设置音频通话的声音质量。

通话类型：记录本次通话的类型，如一对一通话、多人通话等。

网络传输方式：本此通话网络传输的方式

输出要求：

通话状态：返回视频通话的状态，包括通话中、已结束、未接通等状态。

通话时长：记录通话的时长，用于统计和计费等目的。

视频数据：如果需要在通话过程中实时展示视频画面，可以返回视频数据以供显示。

音频数据：如果需要在通话过程中实时播放声音，可以返回音频数据以供播放。

错误信息：如果通话状态为失败或异常，可以返回相应的错误信息，方便定位问题。

丢包率：返回视频通话过程中的丢包率，方便判断系统及网络问题。

## 2.3 运行环境规定

### 2.3.1 设备

处理器型号：Intel Core i5-7500 及以上

内存容量：8G以上

硬盘容量：256G以上

输入输出设备：

摄像头：Logitech C920 HD Pro 联机

麦克风：Audio-Technica ATR2100x-USB 联机

扬声器/耳机：Bose QuietComfort 35 II 联机

显示器：ASUS ProArt PA278QV 联机

数据通信设备：

路由器：TP-Link XDR3010易展版

网络适配器（网卡）：Realtek PCIe GbE Family Controller

功能键：无

其他专用硬件：无

### 2.3.2 支持软件

支持软件：目前市面上常见的浏览器

支持的操作系统：Windows、Mac、Linux、iOS和Android。

# 第3章 系统设计

## 3.1 总体设计

### 3.1.1 相关技术

1、前端技术：HTML、CSS、JavaScript等。

2、后端技术：Node.js、Express框架、typeScript等。

3、实时通信技术：WebSocket等。

4、服务器环境：Nodejs-WebSocke等。

这些技术和环境是制作网络聊天系统所必需的，其中前端技术负责设计界面和用户交互，后端技术负责处理业务逻辑和数据存储，实时通信技术负责保障消息的实时传输，服务器环境负责提供稳定的服务和安全保障。

### 3.1.2总体设计范围

（1）系统架构设计：系统架构设计是系统设计的核心，它涉及到系统的整体框架设计和各个模块之间的关系。在该环节中，需要确定系统的三层架构，即表示层、业务逻辑层和数据访问层，并且需要明确每个层次中的功能模块和接口规范。

（2）用户界面设计：用户界面设计是系统设计的一个重要方面，它影响着用户对系统的第一印象。在该环节中，需要考虑到用户习惯、文化背景和视觉感受等因素，以设计出符合用户期望的交互界面和样式。

（3）系统测试计划：系统测试计划是系统设计的重要环节之一，它涉及到整个系统测试的规范和流程。在该环节中，需要制定出详细的测试计划和测试用例，并且对测试结果进行分析和修正，以确保系统的质量和可靠性。

（4）上线部署方案：系统上线部署方案是系统设计的最后一个环节，它涉及到整个系统的部署和发布流程。在该环节中，需要考虑到服务器配置、带宽限制和数据备份等问题，并且需要确定系统的发布策略和更新计划。

### 3.1.3专业术语

SDP（Session Description Protocol）：会话描述协议，用于描述发起会话的端和接收会话的端之间的媒体参数和网络信息。

本地描述（local SDP）：通话发起者的SDP

远程描述（remote SDP）：被呼叫者的SDP

ICE（Interactive ConnectivityEstablishment）：交互式连接建立，用于在不同的网络之间建立连接。它尝试通过NAT（网络地址转换）等网络层面的障碍，找到双方之间最佳的通信路径。

ICE候选(ICE candidate)：指代ICE建立连接的候选方案之一，用于建立点对点连接

STUN（Session Traversal Utilities for NAT）：用于穿透NAT，使客户端可以发现它们在互联网上的公共IP地址和端口。STUN服务器帮助客户端确定NAT类型并获取公共IP地址，但不能直接在两个设备之间建立连接。

TURN（Traversal Using Relay NAT）：用于在两个设备之间建立连接，即使它们都在NAT后面或防火墙后面，也可以通过TURN服务器中继流量来实现。TURN服务器通常被用作备用方案，因为它们会增加延迟和带宽成本。

信令服务器：（Signaling server），用于在两个设备之间传递SDP和候选项信息，以协商连接的参数。可以使用各种协议来实现信令，包括WebSocket、HTTP和XMPP等。

MediaStream：媒体流，由音频和视频轨道组成的数据流。在WebRTC中，它表示从麦克风、摄像头或其他源收集到的媒体数据。

PeerConnection：用于建立点对点连接的对象。PeerConnection对象通过交换SDP和候选项信息来协商连接参数，并使用ICE框架选择最佳的通信路径。

DataChannel：数据通道，允许在两个设备之间传输任意类型的数据，而不仅仅是媒体数据。数据通道可以用于实现聊天、游戏、文件传输等功能。

### 3.1.4 WebRTC中P2P连接的网络架构

应用层：应用层是WebRTC应用程序的顶层，包括了用户界面、应用逻辑和信令协议的实现。用户可以在界面上选择要进行的实时通信操作，例如发起视频通话、发送消息等。在应用层中，WebRTC应用程序使用信令协议与服务器进行通信，交换SDP和候选项信息，并协商连接参数。

传输层：传输层负责将媒体数据从应用程序发送到远程对等端或从远程对等端接收媒体数据。WebRTC使用RTP（Real-time Transport Protocol）作为传输层协议，它提供了低延迟和高带宽的媒体传输。

媒体层：媒体层包括音频和视频编解码器，用于将原始媒体数据编码为可传输的格式，以及在接收端解码接收到的媒体数据。WebRTC支持多种编解码器，包括VP8、VP9、H.264、Opus等。

NAT遍历层：NAT遍历层用于处理网络地址转换（NAT）和防火墙等网络障碍，以便建立点对点连接。WebRTC使用ICE框架（Interactive Connectivity Establishment）来处理NAT遍历，它尝试通过STUN和TURN服务器找到最佳的通信路径。

信令层：信令层负责在两个对等端之间交换信令消息，以协商连接参数和建立点对点连接。WebRTC应用程序可以使用不同的信令协议，例如WebSocket、HTTP和XMPP等。

网络层：网络层提供了IP地址和端口等网络信息，并负责将数据包从源设备传输到目标设备。WebRTC应用程序使用UDP协议进行数据传输，因为UDP供了低延迟和高带宽的数据传输。

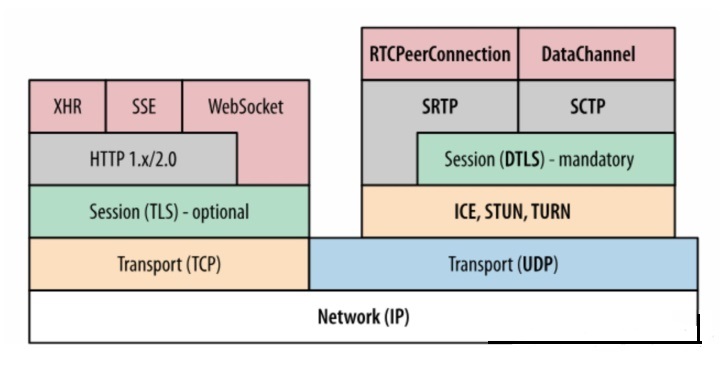


图3-1 WebRTC网络原理图

### 3.1.5 基本设计概念和处理流程

该系统的主要处理流程如下：

初始化：通话系统的初始状态，包括客户端和服务器的初始化设置。

用户身份验证：用户在客户端登录，并通过服务器进行身份验证。

媒体协商：两个客户端之间进行媒体协商，确定音视频编解码器、分辨率、帧率等参数。

信令通信：客户端通过信令服务器进行通信，交换媒体协商信息、ICE候选地址、SDP（会话描述协议）等。

ICE候选地址收集：客户端收集本地的网络地址，包括IP地址、端口号等。

NAT穿透：通过STUN服务器获取本地NAT后面的公网地址，以克服网络中的NAT限制。

建立P2P连接：客户端通过ICE框架，将收集到的ICE候选地址发送给对方，并尝试建立点对点（P2P）连接。

媒体传输：一旦P2P连接建立成功，客户端开始通过P2P连接传输音视频数据。

音视频处理：客户端接收并处理对方发送的音视频数据，包括解码、播放、渲染等操作。

通话控制：客户端支持通话控制功能，例如静音、关闭摄像头、挂断等。

数据传输：除了音视频数据外，客户端还可以通过P2P连接传输其他数据，例如文本消息、文件等。

通话结束：通话结束时，客户端断开P2P连接，释放资源，并进行清理工作。

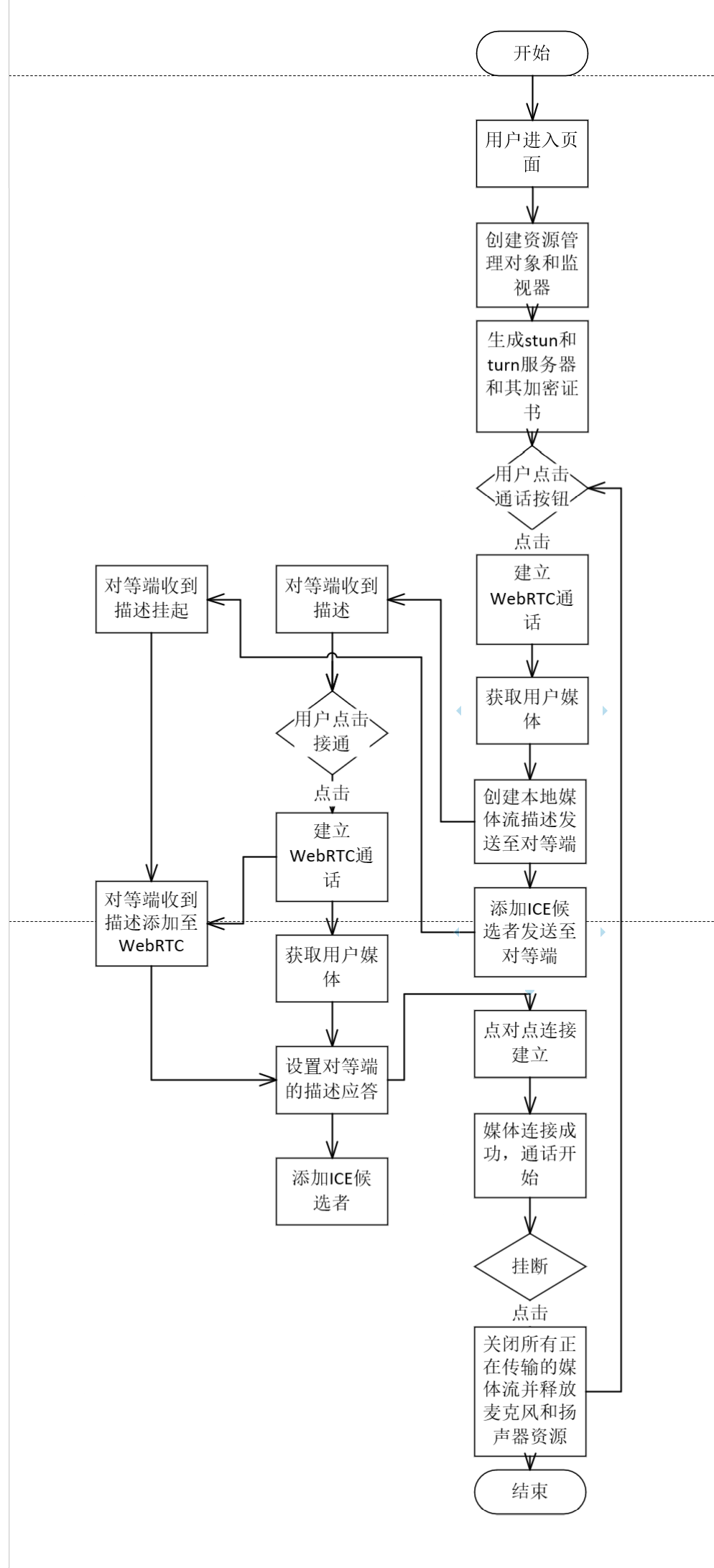


图3-2 视频通话系统流程图

### 3.1.6 结构

系统结构是对一个系统的组成、功能和交互方式进行描述和定义的过程。它为系统开发人员提供了一个指导框架，使他们能够更好地理解系统的整体架构，并在此基础上进行系统设计和实现。

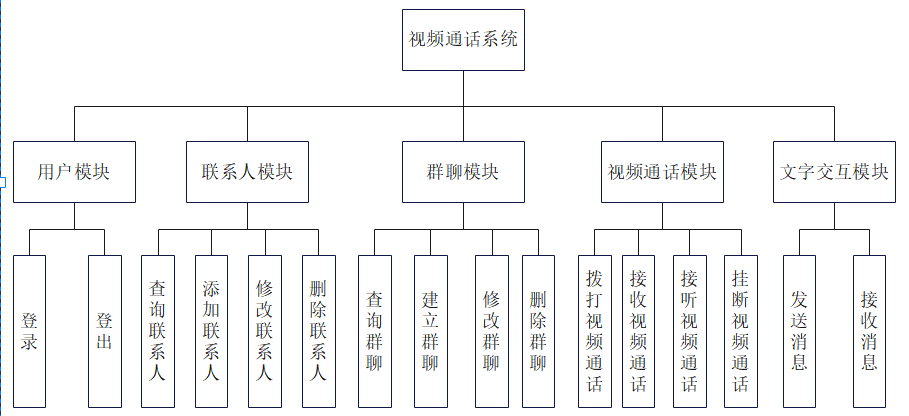


图3-3 视频通话系统功能模块图

### 3.1.7 系统模块介绍

#### 3.1.7.1联系人模块介绍：

作为一个具有视频通话以及文字聊天功能，联系人是必不可少的一部分，具有联系人我们才可以和指定的用户进行聊天沟通，它提供了用户管理联系人、进行端对端通话和文字聊天的功能。通过联系人模块，用户可以方便地添加、编辑和删除联系人，将其他用户的信息保存在联系人列表中。同时，该模块还负责处理好友请求，向被请求的用户发送通知，并在对方同意后将其添加到联系人列表中。

#### 3.1.7.2群聊模块介绍：

它提供了一种便捷的方式让多个用户同时进行文字聊天、分享信息和进行协作。

通过群聊模块，用户可以创建自己的群组或加入已存在的群组。用户可以选择邀请其他联系人或接受群组的邀请，以参与到特定的群聊中。

一旦进入群聊，用户可以在群组中与其他成员实时交流。他们可以发送文字消息、链接等等，并即时收到其他成员的回复。这种实时互动使得群聊成为一种高效的协作工具，能够促进团队之间的有效沟通和合作。

#### 3.1.7.3 视频通话模块介绍：

视频通话模块提供了实时、高质量的双向视音频通信，使用户能够进行面对面的远程交流。

通过视频通话模块，用户可以选择联系人或进入群组，并发起视频通话请求。利用WebRTC技术，该模块能够建立点对点的连接，直接将视频和音频数据传输到对方设备上，实现实时的视音频通信。

在视频通话期间，用户可以同时观看对方的实时视频画面并听到其声音。这种沟通方式使得用户能够更直观地感受到对方的表情、语气和肢体语言，增强了沟通的准确性和亲密性。

此外，视频通话模块也注重用户体验和性能优化。它会根据网络条件和设备性能智能调整视频和音频质量，以保证通话的流畅性和稳定性。同时，该模块也会考虑到安全性和隐私保护，确保通话内容只能被授权的用户访问。

总结而言，视频通话模块是基于WebRTC的P2P视频通话系统中的核心组件。它通过实时、高质量的双向视音频传输，使用户能够进行远程面对面的交流。视频通话模块不仅提供了直观的沟通方式，还注重用户体验和性能优化，以确保通话的流畅性、稳定性和安全性。这样的模块为用户在远距离交流和协作中提供了便捷、高效和真实的体验。

#### 3.1.7.4 文字交互模块介绍：

文字交互模块提供了实时的文字聊天功能，使用户能够方便地进行文字交流、分享信息和表达意见。

通过文字交互模块，用户可以与联系人或群组成员进行实时的文字聊天。他们可以输入文本消息，并即时发送给对方。该模块会将这些消息传输到接收方设备上，让其能够立即看到收到的内容，并进行回复。

### 3.1.8 安全性设计

安全性设计需要考虑到多个方面，包括用户认证、通信加密、数据保护、漏洞修复等。只有在充分考虑到这些因素之后，才能设计出一个稳定、高效、易用和可扩展的基于Web的通话系统，以满足用户和企业的需求。

1.用户认证

用户认证是基于Web的通话系统安全性设计的核心功能之一。该功能需要通过用户名和密码验证用户身份，在确认用户身份后才能访问系统资源和服务。在实现用户认证时，需要采用密码加密技术，以防止用户密码被盗取或破解。同时，还需要考虑到使用Cookie或Session保存用户登录状态，从而避免用户在一段时间内反复输入用户名和密码。

2.通信加密

通信加密是基于Web的通话系统安全性设计的另一个核心功能。该功能需要在用户与服务器之间建立安全连接，并通过加密算法保护通信内容的安全性。在实现通信加密时，需要采用HTTPS协议，以确保通信过程中的安全性。同时，还需要注意选择合适的加密算法和秘钥长度，以提高加密强度和安全性。

3.数据保护

数据保护是基于Web的通话系统安全性设计的另一个重要功能。该功能需要保护用户数据的安全性和隐私性，防止数据泄漏或被非法访问。在实现数据保护时，需要采用数据加密技术，以确保数据在传输和存储过程中的安全性。同时，还需要注意对用户数据进行备份和灾备处理，以防止数据丢失或损坏。

### 3.1.9 功能需求与程序的关系

本条用一张如下的表格说明各项功能需求的实现同各块程序的分配关系。

表3-1 需求程序对应表

|  |  |  |  |
| --- | --- | --- | --- |
|  | call-operation.js | send-message.js | friend-list.js |
| 登录 | × | × | √ |
| 登出 | × | × | √ |
| 查询联系人 | × | × | √ |
| 添加联系人 | × | × | √ |
| 修改联系人 | × | × | √ |
| 删除联系人 | × | × | √ |
| 查询群聊 | × | × | √ |
| 建立群聊 | × | × | √ |
| 修改群聊 | × | × | √ |
| 删除群聊 | × | × | √ |
| 拨打视频通话 | √ | × | × |
| 接收视频通话 | √ | × | × |
| 接听视频通话 | √ | × | × |
| 挂断视频通话 | √ | × | × |
| 发送消息 | × | √ | × |
| 接收消息 | × | √ | × |

## 3.2 详细设计

### 3.2.1 登录界面设计

页面结构：确定页面的整体布局和结构，包括导航栏、页眉、页脚、主体内容区域等。确保页面结构清晰、易于导航和使用。

品牌识别：使用品牌颜色、字体、标志等元素来增强页面的品牌识别度，使用户能够轻松地识别和记住品牌。

响应式设计：确保页面具有响应式设计，能够在不同的设备上自适应调整布局和大小，从而提供更好的用户体验。

内容排版：设计清晰、易读的排版，包括标题、副标题、正文、列表等，使用户能够快速了解页面内容。

视觉效果：使用高质量的图片、图标、动画等视觉元素来增强页面的吸引力和视觉效果。

导航栏设计：设计易于导航和使用的导航栏，确保用户能够轻松找到所需的信息和功能。

表单设计：设计易于填写和提交的表单，确保用户能够轻松完成表单填写和提交。

响应速度：确保页面响应速度快，加载时间短，从而提供更好的用户体验。

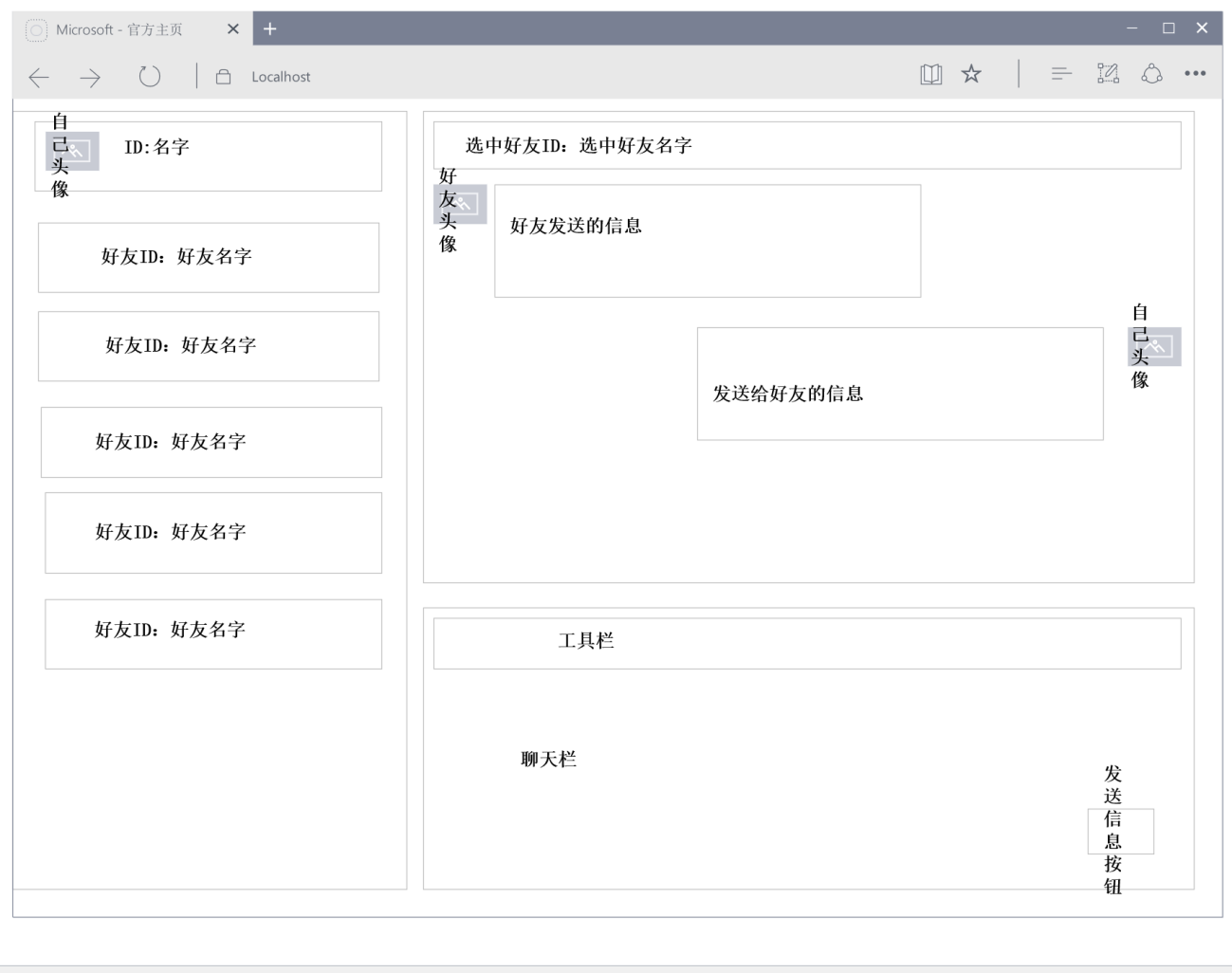


图3-4 前端设计图

### 3.2.2 通信流程

（1）第一阶段：获取媒体

首先，由客户端 A 创建会议房间，客户端 B 加入，信令服务器对相应指令做出反应；

此后客户端 A 和客户端 B 均创建一个 PeerConnection 对象，并使用 addTrack 方法将 getUserMedia 接口获取的媒体数据封装成 MediaStream 流，最后将 MediaStream 流添加到PeerConnection 对象中。

（2）第二阶段：媒体协商

（a）客户端 A 调用 PeerConnection 对象中的 CreateOffer 方法创建一个 SDP 对象，并使用 SetLocalDescription 方法保存 SDP 信息，随后将 SDP 信息发送给信令服务器。信令服务器将 SDP 信息转发给媒体服务器，并与媒体服务器进行协商，创建传输数据流的媒体管道 MediaPipeline 和传输端点 RTCEndpoint。

（b）媒体服务器收到信令服务器发来的 SDP 信息，首先调用远端会话描述

SetRemoteDescription方法将SDP信息保存，并调用ProcessOffer方法获取自身的SDP信息，最后将自己的 SDP 信息发送给信令服务器，由信令服务器转发给客户端 A。

（d）客户端 A 收到信令服务器发来的 SDP 信息，调用 SetRemoteDescription 方法将客户端 B 的 SDP 信息保存。客户端 B 的流程与客户端 A 一致。客户端和媒体服务器在收到SDP 信息后，会根据对方的 SDP 信息来建立相应的视频 Channel 和音频 Channel。

（3）第三阶段：网络协商

（a）客户端 A 和客户端 B 开启 Candidate 数据的收集，与 Coturn 服务器连接并获取自身的 Candidate 数据，Coturn 服务器在获取到数据后会调用 PeerConnection 的 OnIceCandidate接口来通知客户端 A。

（b）客户端端 A 和客户端 B 会将 Candidate 数据发送给信令服务器，由信令服务器转发给媒体服务器。

（c）媒体服务器通过 AddIceCandidate 方法保存客户端的 Candidate 数据，并将自己的

Candidate 数据通过信令服务器发送给客户端，客户端通过同样的方法保存数据。

（4）第四阶段：音视频通信

客户端 A 和客户端 B 通过会话管理机制创建并加入了会议房间，完成了各自的媒体协商和网络协商，媒体服务器获取了双方的 SDP 信息和 Candidate 数据，并建立了与客户端之间端到端的传输通道，可以互相发送和接受音视频流。当客户端 A 收到音视频流后会通过 PeerConnection 的 OnAddStream 接口返回一个标识为客户端 B 的 MediaStream 对象，并通过 Video 标签在页面进行渲染或显示。

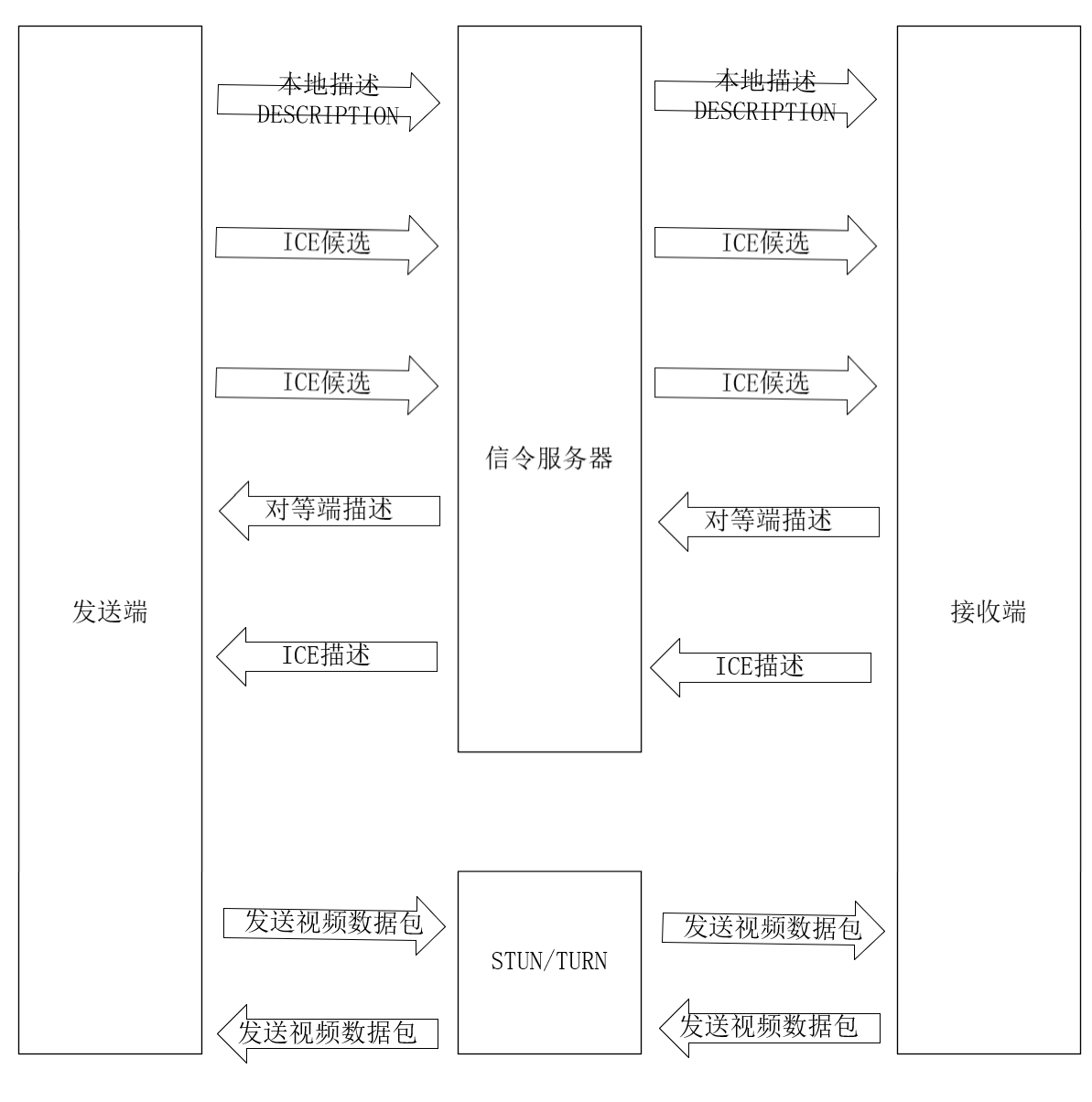


图3-5 通信流程图

### 3.2.3 信令服务器设计

客户端之间进行数据传输，在传输之前要建立一个信令机制用以协调控制通

信。WebRTC 没有标准化信令协议，因此本系统可以根据实际需要选择信令协议和 通信信道，这能够有效避免信息冗余。具体来看，信令主要有以下四点作用：

(1) 媒体协商：在进行会话之前需要配置诸如音频、视频等媒体类型的数据，

这些配置数据，存在于 SDP 当中。SDP 中也包括编解码器的类型，可以通过选择来指定编解码器类型，SDP 中还包括这些编解码器设置参数的信息，这些功能的

完成都是需要通过建立连接的浏览器进行 SDP 信息交换来实现的。

(2) 标识和验证通讯者身份：信令信道会提供会话者身份可以用来选择认证。

(3) 控制媒体会话：媒体会话控制应该包括会话从建立到结束整个的流程的功

能控制以及指示当前媒体会话的状态，但是基于 WebRTC 的信令只需要其完成会 话的发起和更改便可。

双占用解决：视频会话中可能存在通信双方同时发起或者更改会话的情况，WebRTC 信令会采用 SDP 消除多种双占用情况，从而减少因为双占用导致的出现如会话无法确定等问题的发生。WebSocket 是一种全双工的信令传输协议。图 2.4 展示了 WebSocket 的传输模式。WebSocket 必须拥有公共 IP 地址，运行 HTTP 服务才能被访问。当前主流的浏览器都支持 WebSocket。

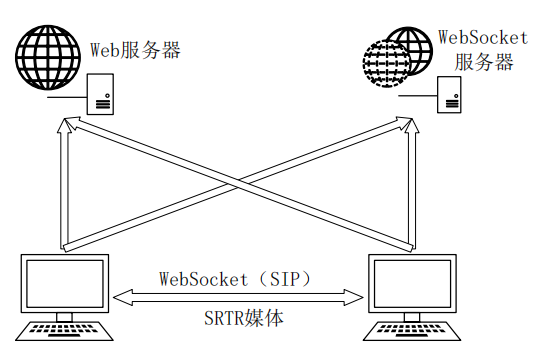


图3-6 WebSocket信令服务器连接图

本文所做的视频会议系统选用 WebSocket 作为信令协议。WebSocket 于2011 被确定为国际标准，WebSocket 信令协议是基于 TCP 的全双工协议，但是不同于 TCP 的三次握手建立连接，WebSocket 建立通信通道仅需要一次握手即可便能建立连接，之后客户端之间就可以自由传递信息，客户端会保持监听状态，除非客户端或者服务器断开 Websocket 连接，否则不需要再轮询服务器。

在一个视频会议系统中，信令服务器要接收各种客户端的操作，并做出处理。 而加入，与他人聊天，与他人通话以及退出是一次软件工作的完整流程，因此每 有这四种操作发生时，系统便会向信令服务器发送信号。加入新成员对应MessageType.NEW\_FRINTD命令，发送信息对应MessageType.MESSAGE命令，视频通话对应MessageType.DESCRIPTION，MessageType.REMOTE，MessageType.CANDIDATE命令，退出对应MessageType.CLOSE命令。收到信号后信令服务器便会根据信令内容做出调度处理。这三种信令是聊天系统系统最重要的信令。其具体的响应过程如图 3-7 所示。

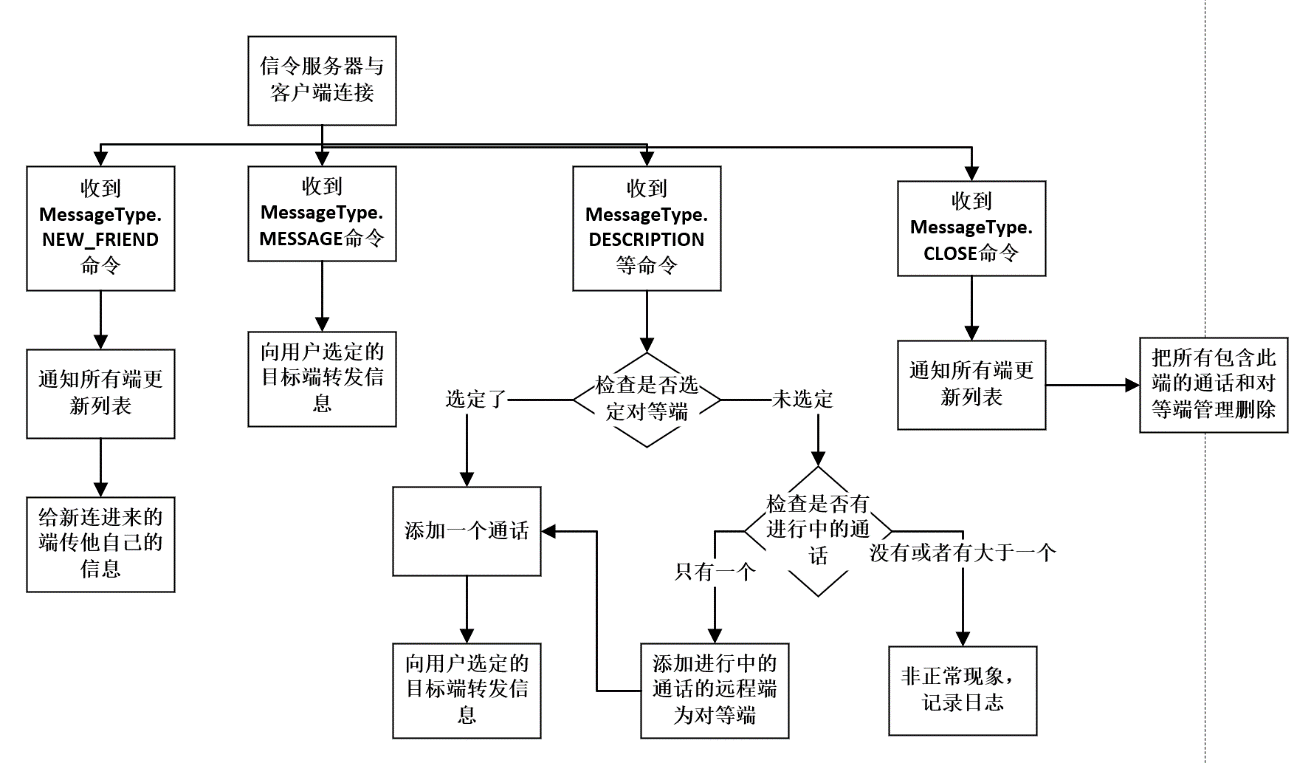


图3-7 信令服务器流程图

在视频会议系统中，信令服务器除了对上述四种最重要的命令做出响应外，其他包括用户操作通话记录联系人，用户修改昵称等都需要做出响应。这些信令我们称之为普通信令。

### 3.2.3 接口设计

在该系统开发过程中用到了很多的WebRTC接口，为更加明确的表达出来故使用表格呈现。

表3-2 WebRTC接口对应表

|  |  |  |
| --- | --- | --- |
| API |  | 功能 |
| MediaStream API | getAudioTracks | 返回一个包含音频轨道的数组 |
| getVideoTracks | 返回一个包含视频轨道的数组 |
| addTrack | 将一个音频或视频轨道添加到媒体流中 |
| removeTrack | 从媒体流中删除一个音频或视频轨道 |
| RTCPeerConnection API | createOffer | 创建一个包含本地媒体流和连接选项的SDP描述符 |
| createAnswer | 创建一个包含远程媒体流和连接选项的SDP描述符 |
| setLocalDescription | 设置本地SDP描述符 |
| setRemoteDescription | 设置远程SDP描述符 |
| addIceCandidate | 添加一个ICE（交互式连接建立）候选项 |
| onicecandidate | 当ICE候选项可用时触发的事件处理程序 |
| RTCDataChannel API | send | 将数据发送到远程浏览器 |
| onmessage | 当接收到数据时触发的事件处理程序 |
| onopen | 当数据通道打开时触发的事件处理程序 |
| onclose | 当数据通道关闭时触发的事件处理程序 |
| RTCIceServerAPI | urls | 服务器的URL或URL列表 |
| username | 可选的用于身份验证的用户名 |
| credential | 可选的用于身份验证的密码 |
| RTCStats API | getStats | 返回一个包含与选择器匹配的统计信息的 Promise。选择器可以是null，表示返回所有统计信息，或者是一个包含要筛选的统计信息类型的对象 |
| MediaDevices API | getUserMedia | 请求访问音频和视频设备的权限，并返回包含MediaStream的 Promise |
| enumerateDevices | 返回一个包含可用设备的数组，每个设备包含deviceId、kind和label属性 |

### 3.2.4 对象类设计

前端类的核心是CallOperation。这个类与多个类有着依赖以及直接关联的关系。其中与这个类最密切的直接关联是APPGlobal类，这个类是前端的核心引用类。APPGlobal类与CallTimer类有直接关联，并且实现了CallTimer的接口。同时实现了friendlistExport接口。CallOperation类同时与ConstantObserver类有着直接关联的关系，因为内部调用了多个这个类的实例对象。于CallingState枚举类有着直接关联的关系，因为CallOperation类中需要直接判定callingState枚举类的实例。与ObserverCallBackType，MessageType以及MessagePacket类有着依赖关联的关系，因为需要间接的调用其实例对象。

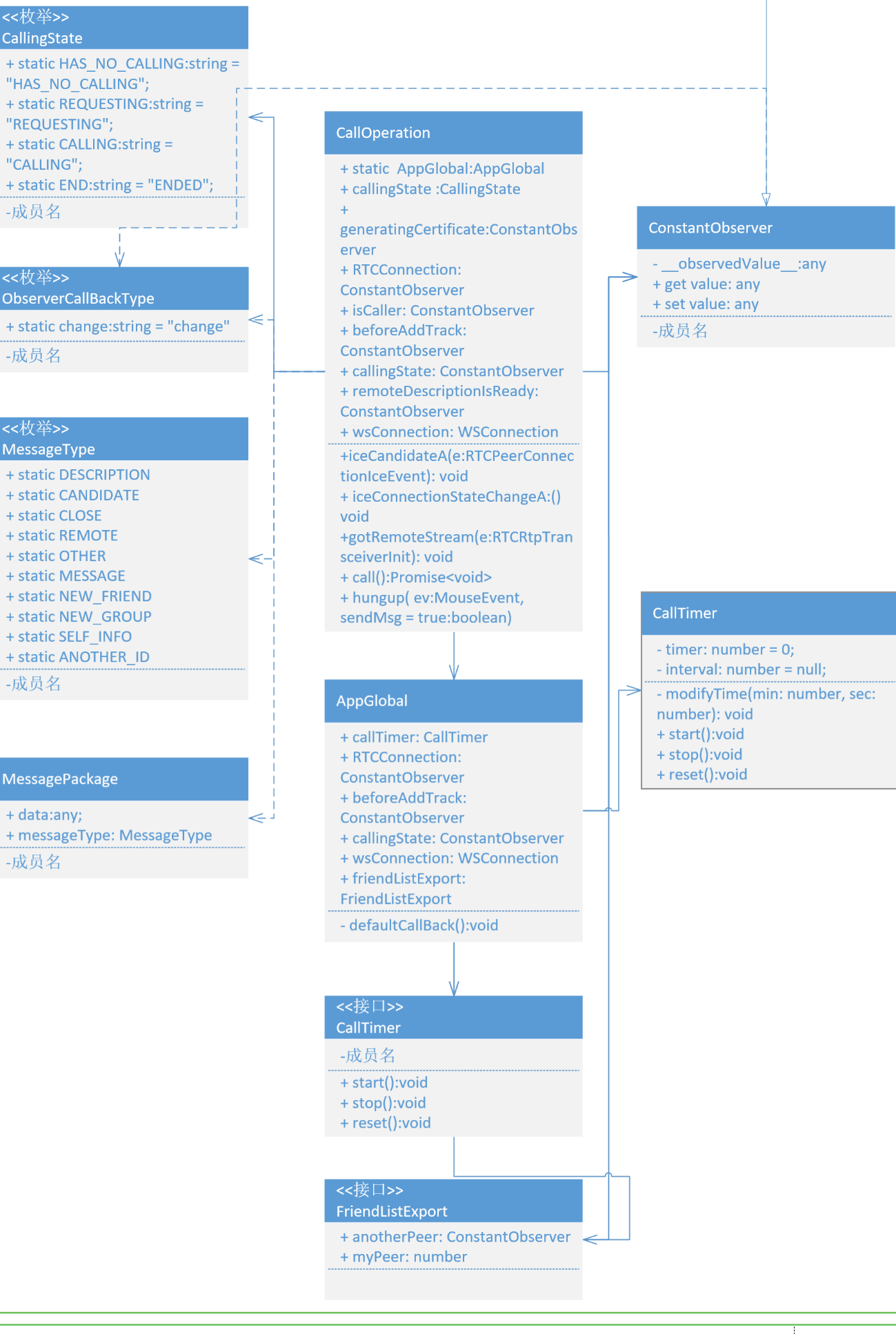


图3-8前端类图

后端类的核心是Peer2peerService类。Peer2peerService类对着其他三个类有依赖关联关系，因为其间接的使用了这几个类的实例对象进行端对端的管理工作。

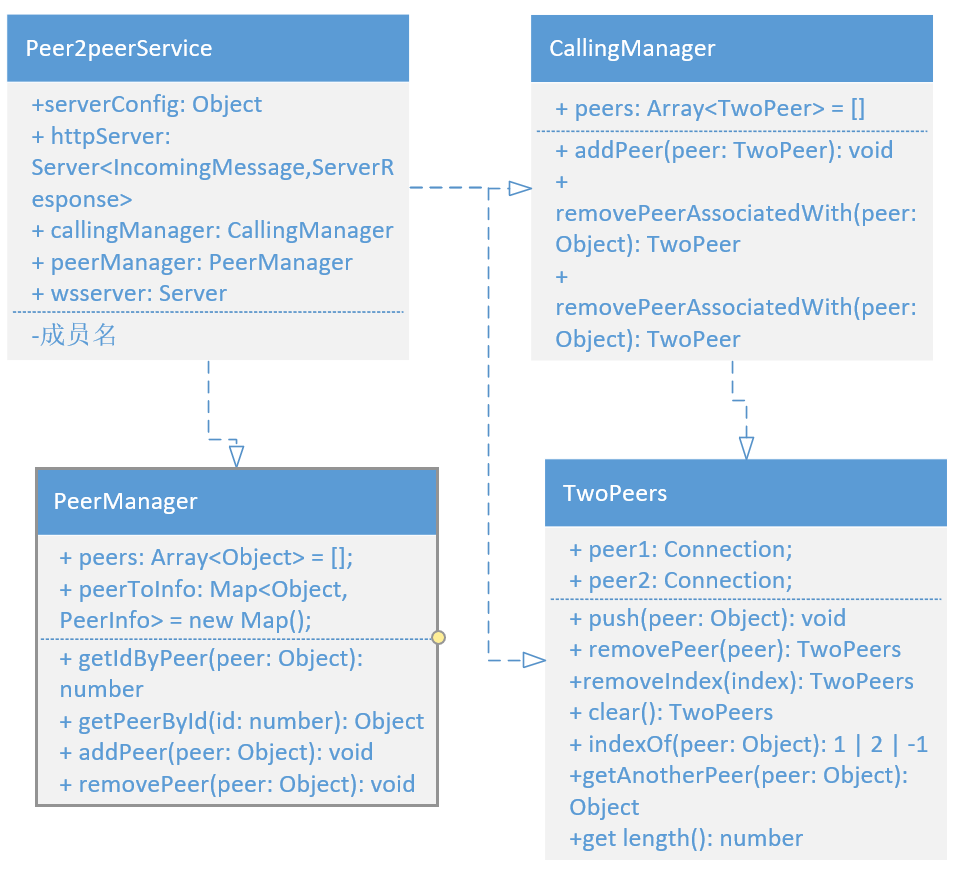


图3-9 后端类图

### 3.2.5 类描述

**MessageType枚举：**

信息的类型的枚举

静态属性：

static number DESCRIPTION = 1，WebRTC创建的本地描述

static CANDIDATE = 2，WebRTC创建的ICE候选

static CLOSE = 3，挂断信号

static REMOTE = 4，WebRTC创建的远程描述

static OTHER = 5，WebRTC创建的其他不可控信息

static MESSAGE = 6，文字信息类型

static NEW\_FRIEND = 7，刷新好友列表请求类型

static NEW\_GROUP = 8，刷新群组

static SELF\_INFO = 9，用于服务器给客户发送他们自己的信息

static ANOTHER\_ID = 10，用于向服务器发送选中对等端的ID

**MessagePackage类：**

用于封装WebSocket传递信息的信息包，里面有需要的最基本的信息

实例属性：

any messageType，发送的消息的类型，一般是字符串或者是数字或者是Symbol

any data，发送消息的text实体，是网络信息的储存位置，一般是一个字符串

**MessageMap类：**

这个类用于处理WebSocket传递过来信息对诸多回调函数的映射

实例属性：

1. WebSocket connection，用于保存一下WebSocket API的实例对象，也就是我们在代码中建立的连接的实例对象

2. Map messageMap，是一个映射的实例对象，用于对消息的类型进行解析后，执行对应的回调函数，例如WebSocket实例对象收到一个Message类型的数据包，就会执行用户注册的Message类型的回调函数。

实例方法：

1. void defaultEvent()，是WebSocket API实例对象收到WebSocket服务器发送的消息的时候默认触发的message事件处理函数，负责将服务器的消息解析后按照Message.prototype.messageType属性调用映射中对应的回调函数

2. void addMap(any type,Function callback)，用于添加消息类型对回调函数的映射，一个类型对应多个回调函数

3. void removeMap(any type)，用于移除一个消息类型对应的所有回调函数

4. void invoke(any type,MessagePackage)，用于调用一个消息类型对应的所有回调函数

5. void refreshEvent(Function eventFun)，是手动添加WebSocket的message事件回调函数的方法，参数默认值是this.defaultEvent

**Message类**

收到消息的后的对于消息的储存模型

实例属性：

1. any id，发出信息者的ID

2. any data，发出的信息

**MessageManager类**

管理客户端收到的消息的类

继承链：Object -> EventTarget -> MessageManager

实例属性：

1. #peerToMessages: Map<number, Array<Message>> = new Map()，一个映射，从一个好友的ID映射到其所有消息

实例方法：

1. Array<Message> getMessages(peer: number)，根据发送目标的ID获取所有消息

2. void addMessages(peer: number, message: Message)，用向peerToMessages映射添加消息信息，并触发change事件，第一个参数是指想要保存的信息的发送者的ID，第二个参数是信息的实例对象

**ConstantObserver类**

一个基于回调函数的监视器类，将待监视的值存入属性，监听属性的变化触发对应事件

继承链：Object -> EventTarget -> ConstantObserver

实例属性：

1. any #\_\_observedValue\_\_，被监视的属性

2. void get value()，获取被监视属性当前的值

3. void set value(any value)，设置被监视属性当前的值,设置后触发change事件

**ObserverCallBackType枚举**

ConstantObserver类的事件类型枚举

静态属性：

1.static string change = "change"，监控数值发生改变的事件

**CallingState枚举**

表示通话状态的枚举

静态属性：

1. static HAS\_NO\_CALLING = "HAS\_NO\_CALLING"，当前没有通话

2. static REQUESTING = "REQUESTING"，通话请求中

3. static CALLING = "CALLING"，通话中

4. static END = "ENDED"，通话结束

**CallTimer类**

通用计时器

实例属性：

1. number #timer = 0，表示当前计时器start()方法调用后所经过的秒数

2. nummber #interval，表示当前所执行的计时器实例的编号

实例方法：

1. void #modifyTime(number min,number sec)，根据传入的分钟和秒数信息修改计时器对应的HTML标签

2. void start()，开始计时，并每隔一秒调用一次#modifyTime()方法，如果已经有计时器实例的话则什么也不执行

3. void stop()，停止计时，删除计时器实例对象

4. void reset()，重置时间，将#time属性设置为0，并调用#modifyTime()方法

**PeerInfo类**

临时管理用户个人信息的类，类似于用户信息的模型

实例属性：

1. number id，用户ID

2. string name，用户名

实例方法：

1. string getInfo()，返回一个可以概括用户所有信息的数组

静态方法：

1. string ganerateName()，随机生成一个用户名

**TwoPeer类**

封装一组P2P的类，提供了一些方法可以在两个端之间快速操作

实例属性：

1. Peer peer1，两个端的第一个端

2. Peer peer2，两个端的第二个端

3. number get length()，获目前已经存好了几个端

实例方法：

1. void push(Peer peer)，添加一个端，如果第一个是空着的就存到第一个端的位置上，第二个同理

2. TwoPeers removePeer(Peer peer)，删除指定的端

3. TwoPeers removeIndex(nummber index)，删除指定索引的端

4. TwoPeers clear()，删除所有端

5. 1|2|-1 indexOf(Peer peer)，获取当前端是第一个还是第二个还是两个都不是

6. Peer getAnotherPeer(Peer peer)，获取两个端中的另一个端

**PeerManager类**

一个WebSocket端对服务器给他分配信息的管理类

实例属性：

1. peers: Array<Peer> = []，被管理的所有Peer实例对象的合集

2. peerToInfo: Map<Peer, PeerInfo> = new Map()，一个从Peer实例对象到这个实例对象对应的PeerInfo实例对象的映射

实例方法：

1. number getIdByPeer(Peer peer)，通过Peer实例对象获取他的ID

2. Peer getPeerById(number id)，通过ID获取对应的Peer实例对象

3. void addPeer(Peer peer)，向管理中添加一个Peer实例对象，并自动建立一个这个Peer实例对象到一个PeerInfo实例对象的映射

4. void removePeer(Peer peer)，从管理中删除一个Peer实例对象，并从映射中删除与这次对应的PeerInfo实例对象

**CallingManager类**

用于管理通话中的两个端的管理类，封装了一些用于简化操作的方法

实例属性：

1. Array<TwoPeer> peers = []，保存目前所有正在通话中的两个端

实例方法：

1. void addPeer(TwoPeer peer)，添加一个TwoPeer实例对象，也就是添加一对Peer

2. TwoPeer removePeerAssociatedWith(Peer peer)，检索所有已经添加了的TwoPeer，如果他们的两个端中有一个和参数传过来的peer一致的话，就删除这个TwoPeer实例对象

3. TwoPeer getPeerAssociatedWith(Peer peer)，检索所有已经添加了的TwoPeer，如果他们的两个端中有一个和参数传过来的peer一致的话，就返回这个TwoPeer实例对象

### 3.2.6 性能

实时音视频通讯从终端的信息采集再到终端的播放，都要经过以下几个流程：

终端信息采集，前处理，编码，传输，解码，后处理，终端播放。 这其中，每一个步骤都会产生一定的延时。按照延时所处的阶段的不同，将其大致分为三类：

T1：端设备延时。

T2：服务器到端的延时。

T3：服务器间的延时。

编解码阶段产生的延迟是造成的视频会议系统延迟的主要原因，而当下我们的计算机性能普遍较好，编解码运算所造成的时延微乎其微，在这一阶段主要延时是缓冲区所造成的。缓冲区又分为编码缓冲区和解码缓冲区。编解码缓冲区可以短暂的存储一定的数据信息，有效的平滑当前码率波动的问题，缓冲区的大小直接决定了其所能承受的码率的波动范围，缓冲区越大，越能承受码率波动，其容错能力越好，但是缓冲区越大造成的延迟也越大。

下图为利用性能测试工具测算出的延迟，做了多次实验，t1代表开始时间，t2代表结束时间，△t代表延迟，单位为毫秒。

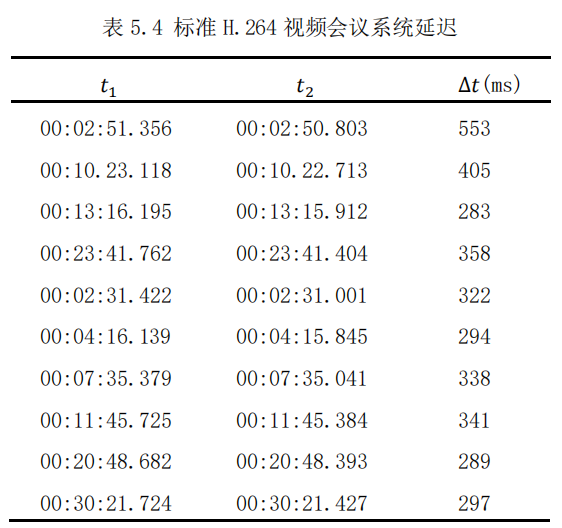


图3-10 延迟统计图

# 第4章 系统实现

## 4.1 基础平台及资源

### 4.1.1 PHPSTORM IDE

PHPSTORM是一款专业的PHP集成开发环境，它提供了强大的代码编辑、调试、测试、重构、版本控制等功能，让PHP开发者能够高效地编写高质量的代码。PHPSTORM支持多种PHP框架，如Laravel, Symfony, Zend Framework, Yii, CakePHP等，以及前端技术，如HTML, CSS, JavaScript, TypeScript, React, Vue.js等。PHPSTORM还集成了Composer, PHPUnit, Xdebug, Docker, Vagrant等工具，帮助开发者构建和部署应用。PHPSTORM的优势有以下几点：

智能代码提示：PHPSTORM能够根据上下文和项目结构，提供合适的代码补全、参数提示、文档查看等功能，让开发者快速输入代码，避免错误。

总之，PHPSTORM是一款功能强大、易用、高效的PHP集成开发环境，它能够满足不同类型和规模的PHP项目的开发需求，是PHP开发者的不二之选。

### 4.1.2 WebRTC API

WebRTC（Web Real-Time Communication）是一组开源的API，它让浏览器和移动应用能够实现实时的音视频通话和数据传输，无需安装任何插件或第三方软件。WebRTC的目标是让任何设备，任何平台，任何浏览器都能够进行实时通信，打造一个开放、安全、高效的通信网络。WebRTC的优势有以下几点：

开放和标准：WebRTC是由W3C和IETF共同制定的标准，它遵循开放的原则，让任何人都能够使用和贡献。WebRTC也得到了主流浏览器的支持，如Chrome, Firefox, Safari, Edge等，让用户无需下载或更新任何软件，就能够享受实时通信的体验。

安全和私密：WebRTC使用了端到端的加密技术，保证了通信的安全性和私密性。WebRTC还使用了STUN, TURN, ICE等协议，帮助用户穿透防火墙和NAT，建立稳定的连接。WebRTC还要求用户在使用音视频设备之前，必须给予明确的授权，保护了用户的隐私权。

高效和灵活：WebRTC使用了最先进的编解码器，如VP8, VP9, H.264, Opus等，提供了高质量的音视频通话。WebRTC还支持自适应码率，根据网络状况动态调整音视频质量，提高通信效率。WebRTC还提供了灵活的API，让开发者能够根据自己的需求，定制通信逻辑和界面。

总之，WebRTC是一组创新的API，它让实时通信变得更加简单、便捷、普及。WebRTC已经被广泛应用于社交、教育、娱乐、医疗、企业等领域，为用户带来了更加丰富和互动的通信体验。

### 4.1.3 WebSocket API

描述数据加载的必要性和意义。WebSocket是一种基于TCP协议的网络通信协议，它允许在客户端和服务器之间建立持久性的双向通信连接，以实现实时数据传输和交互。WebSocket API是一组在浏览器中使用WebSocket协议的接口，它们允许JavaScript应用程序通过WebSocket与远程服务器进行通信。

### 4.1.4 Node.js

Node.js是一种基于Chrome V8引擎的JavaScript运行时环境，它让JavaScript能够在服务器端运行，实现高性能的网络应用。Node.js的特点有以下几点：

事件驱动和非阻塞IO：Node.js使用了事件循环和回调函数的模式，让服务器能够处理大量的并发请求，而不需要创建多线程或多进程。Node.js也使用了非阻塞IO，让服务器能够异步地读写文件、数据库、网络等资源，提高了吞吐量和响应速度。

总之，Node.js是一种现代的JavaScript运行时环境，它让JavaScript能够在服务器端发挥其强大的潜力，实现高性能、高并发、高可扩展的网络应用。Node.js已经被广泛应用于Web开发、API开发、微服务、实时通信等领域，为用户带来了更加快速和流畅的网络体验。

### 4.1.5 nodejs-websocket

描述数据加载的必要性和意义。Node.js WebSocket是一个基于Node.js平台的WebSocket实现，它提供了一组简单易用的API，可以轻松地实现实时数据传输和交互。Node.js WebSocket基于WebSocket协议和Node.js的事件驱动模型实现，可以实现高效的双向通信，适用于实时性要求较高的应用场景。

以下是Node.js WebSocket的介绍：

WebSocket是一种基于TCP协议的网络通信协议，它允许在客户端和服务器之间建立持久性的双向通信连接，以实现实时数据传输和交互。WebSocket协议通过握手协议和数据传输协议实现，握手协议用于建立WebSocket连接，数据传输协议用于实现双向数据传输。

Node.js WebSocket协议支持使用TLS/SSL加密来保证连接的安全性。在使用加密的WebSocket连接时，需要使用以“wss://”开头的URL地址，同时需要在服务器上安装有效的SSL证书。使用加密的WebSocket连接可以有效地保护数据的隐私和完整性。

总之，Node.js WebSocket是一个基于Node.js平台的WebSocket实现，它提供了一组简单易用的API，可以轻松地实现实时数据传输和交互。Node.js WebSocket基于WebSocket协议和Node.js的事件驱动模型实现，具有高效性、简单易用和跨平台性等优势。使用Node.js WebSocket，可以轻松地创建WebSocket服务器和客户端，并处理WebSocket连接的各种状态和消息。同时，Node.js WebSocket还支持文本和二进制数据类型的消息，以及使用TLS/SSL加密保证连接的安全性。Node.js WebSocket适用于实时性要求较高的应用场景，例如在线游戏、实时监控和即时通讯等。

### 4.1.6 Google标准STUN/TURN服务器

STUN（Session Traversal Utilities for NAT）和TURN（Traversal Using Relays around NAT）是两种用于实现NAT穿透的协议，它们让位于不同网络的用户能够建立点对点的连接，进行实时的音视频通话和数据传输。STUN和TURN的作用有以下几点：

STUN：STUN是一种简单的协议，它让用户能够通过一个STUN服务器，获取自己的公网IP地址和端口号，以及自己所处的NAT类型。这些信息可以帮助用户与其他用户交换候选地址，尝试建立直接的连接。STUN适用于大多数情况，但是在一些复杂的NAT环境下，如对称型NAT，STUN无法工作。

TURN：TURN是一种备选的协议，它让用户能够通过一个TURN服务器，将自己的音视频流或数据流转发给其他用户，实现间接的连接。TURN可以解决STUN无法处理的情况，但是它会增加服务器的负载和网络的延迟。

总之，STUN和TURN是两种互补的协议，它们让实时通信变得更加可靠和普遍。STUN和TURN通常与ICE（Interactive Connectivity Establishment）协议一起使用，让用户能够根据不同的网络状况，选择最优的连接方式。STUN和TURN已经被广泛应用于WebRTC等实时通信技术中，为用户带来了更加顺畅和高质量的通信体验。

## 4.2 主页面

### 4.2.1 主页面实现

这个页面实现了一个基于WebRTC技术的实时通信应用的通用聊天页面，包括呼叫视频电话、文本聊天等功能。下面将对这段代码的实现过程进行详细介绍。

整体页面

在HTML文件中，整个应用被包裹在一个<div>元素中，其id为“container”。该元素分为两个区域，分别是个人信息栏和应用主页面。个人信息栏使用<aside>元素进行定义，其中包括个人头像、昵称和好友列表。应用主页面使用<main>元素进行定义，包括好友信息栏、对话窗口和发送信息栏。对话窗口又由<section>元素包含，其中包含了聊天记录的<ul>列表。文本输入框的区域使用了id为“messagebox-container”的<div>元素进行定义，方便后续的JavaScript操作。最后，定义了一个id为“aside-controller”的<div>元素，用于控制好友信息栏的折叠和展开。该元素使用了position:fixed属性，将其固定在页面底部，方便用户操作。

接下来，在主页面的作用域中定义了一些CSS变量，以便后续样式的使用。这些变量包括了应用的高度和宽度、过渡时间、主要颜色、消息对话框的最大高度、字体颜色等。这些变量可以在整个应用中使用，方便了样式的管理和修改。

然后，定义了一些全局样式，包括了字体、边框、字体颜色等。这些样式将应用于整个页面，方便了整体统一的样式管理。

定义了消息对话框的样式。其中，使用了calc()函数计算了消息对话框的最大高度，以便在消息记录较多时进行滚动。同时，使用了CSS Grid布局实现了消息对话框的布局，包括了消息记录列表、文本输入框等。

同时，使用了CSS变量和过渡效果，使其在鼠标悬停时能够显示出相应的效果。

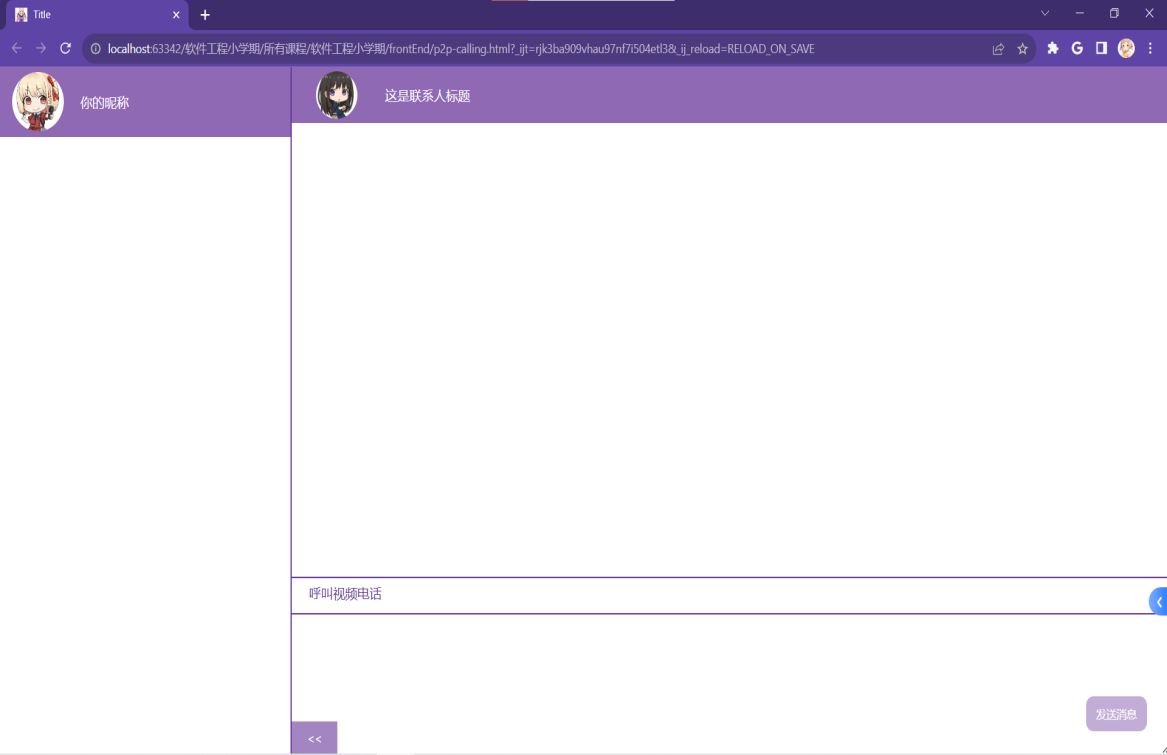


图4-1 主页面

### 4.2.2 前端功能实现

**好友列表**

注册处理MessageType为MessageType.NEW\_FRIEND的MessagePackage的映射函数，首先，它清空了friendList元素的内容。

然后，对于messagePackage.data数组中的每个元素info，它创建一个新的<li>元素，并将info的值作为其文本内容。然后，它使用dataset属性将info的值存储在peerInfo自定义数据属性中。

接下来，它使用querySelectorAll方法选择所有具有 “friend” 类的元素，并将其转换为数组。然后，它遍历该数组中的每个元素friend。

对于每个friend元素，当被点击时，它将friend的文本内容赋值给friendNickName元素的文本内容。它还从friend的peerInfo自定义数据属性中提取冒号之前的部分，并将其转换为数字形式，然后将结果赋值给AppGlobal.friendListExport.anotherPeer.value。

最后，它使用AppGlobal.wsConnection.sendMessage方法向服务器发送一个经过JSON格式化的MessagePackage对象，其中包含AppGlobal.friendListExpor t.another Peer.value和MessageType.ANOTHER\_ID。

总结来说，这段代码的作用是根据输入的信息创建一个好友列表，并为每个好友项添加点击事件，以便在点击时执行一些操作，例如更新某个元素的文本内容，并向服务器发送一条消息。

代码大致如下：

friendList.innerHTML = "";

messagePackage.data.forEach(info => {

let item = document.createElement("li");

item.innerText = info;

item.dataset.peerInfo = info;

item.classList.add("friend");

friendList.appendChild(item);

});

const friends = Array.from(friendList.querySelectorAll(".friend"));

friends.forEach(friend => {

friend.onclick = function () {

friendNickName.innerText = friend.innerText;

AppGlobal.friendListExport.anotherPeer.value = friend.dataset.peerInfo.substring(0, friend.dataset.peerInfo.indexOf(":")) - 0;

AppGlobal.wsConnection.sendMessage(JSON.stringify(

new MessagePackage(AppGlobal.friendListExport.anotherPeer.value, MessageType.ANOTHER\_ID)

));

};

})

注册处理MessageType为MessageType.SELF\_INFO的MessagePackage的映射函数，代码大致如下：

nickname.innerText = messagePackage.data;

AppGlobal.friendListExport.myPeer = messagePackage.data.substring(0, messagePackage.data.indexOf(":")) - 0;

发送信息

实例化MesssageManager类，给发送信息按钮注册事件处理函数，首先，代码检查AppGlobal.friendListExport.anotherPeer.value转换为数字后是否为 NaN。如果是NaN，则退出函数。

接下来，代码获取messagebox的值，并将其赋值给变量content。如果content为空字符串，则退出函数。

然后，清空messagebox的值，并向服务器发送一个经过 JSON 格式化的MessagePackage对象，其中包含content和MessageType.MESSAGE。

接着，代码调用messageManager的addMessages方法将AppGlobal.fri endListExport.anotherPeer.value转换为数字后作为接收者的ID，以及一个包含发送者和内容的Message对象添加到消息管理器中。

接下来，代码注册一个事件监听器，当用户按下回车键时，会触发sendMessageBtn.click()来执行发送消息。

然后，代码注册一个处理MessageType.MESSAGE类型的MessagePackage的映射函数，该函数执行messageManager.addMessages(messagePackage.from - 0, new Message(messagePackage.from, messagePackage.data))。

接下来，代码注册一个事件监听器，当messageManager实例对象的change事件被触发时，会执行以下操作：

首先，清空chatbox元素的内容。

然后，根据AppGlobal.friendListExport.anotherPeer.value转换为数字后获取消息管理器中的消息，将结果赋给变量msgs。如果msgs不存在，则退出函数。

调用scrollIntoView方法，使页面滚动到刚添加的消息元素的位置，实现自动滚动效果。

代码如下：

if (Number.isNaN(AppGlobal.friendListExport.anotherPeer.value - 0)) return;

let content = messagebox.value;

if (content === "") return;

messagebox.value = "";

AppGlobal.wsConnection.sendMessage(JSON.stringify(

new MessagePackage(content, MessageType.MESSAGE)

));

messageManager.addMessages(AppGlobal.friendListExport.anotherPeer.value - 0, new Message(AppGlobal.friendListExport.myPeer, content))

给用户点击回车注册事件，代码如下：

if (ev.key === "Enter") {

sendMessageBtn.click();

}

注册处理MessageType为MessageType.MESSAGE的MessagePackage的映

射函数，此函数执行：

messageManager.addMessages(messagePackage.from - 0, new Messa ge(message Package.from, messagePackage.data))

注册处理MessageManager类实例对象的change事件的函数，此函数执行：

chatbox.innerHTML = "";

let msgs = messageM anager.getMessa ges(Number.parse Float(AppGlobal.frie ndListExport.anotherPeer.value));

if (!msgs) return;

msgs.forEach(msg => {

const li = document.createElement("li");

if (msg.id === AppGlobal.friendListExport.anotherPeer.value - 0) {

li.className = "friends-message";

li.innerHTML = `

<img src="./image/logo2.jpg" alt="no image"/>

<p>${msg.data}</p>;

} else if (msg.id === AppGlobal.friendListExport.myPeer) {

li.className = "my-message";

li.innerHTML = `

<img src="./image/logo.jpg" alt="no image"/>

<p>${msg.data}</p>;

}

chatbox.appendChild(li);

li.scrollIntoView({behavior: "smooth"});

});

对等端ID用一个ConstantObserver实例对象监听，注册一个对等端ID变化的change事件处理函数，此函数流程为 调用MessageManager类实例对象的change事件的函数

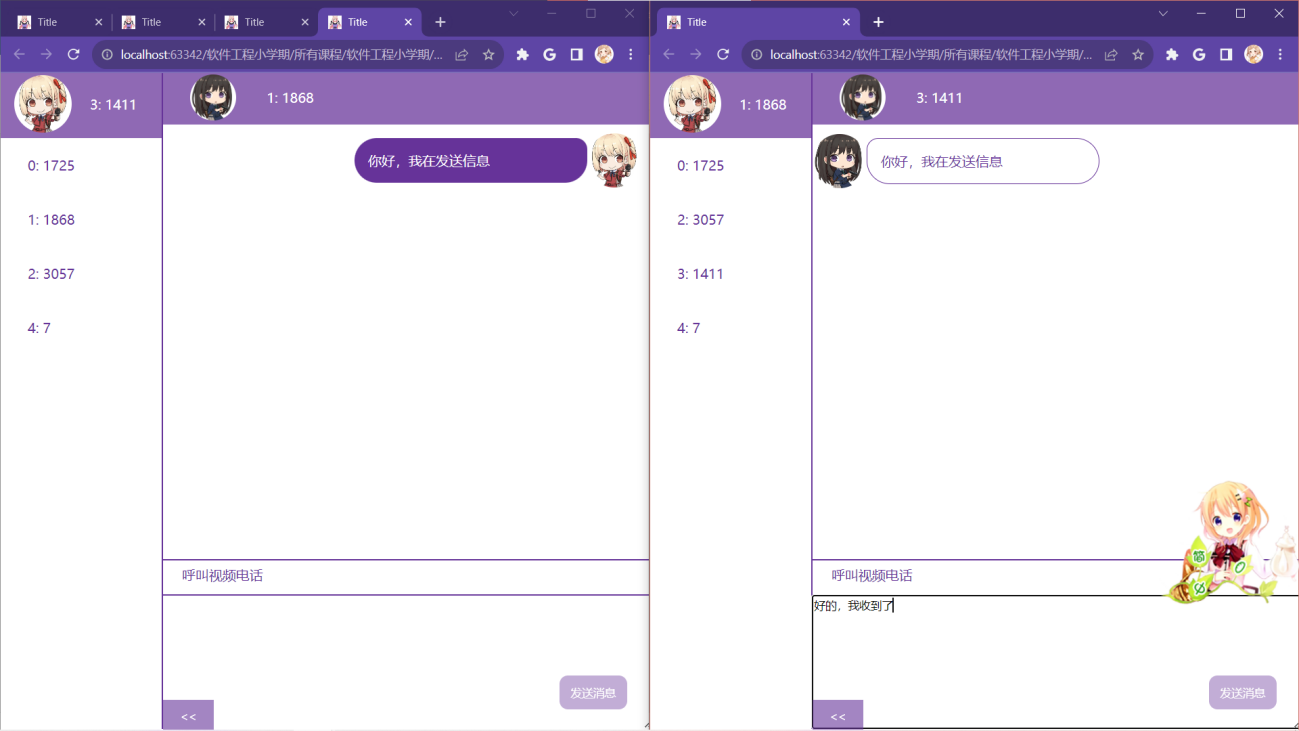


图4-2 发送信息及接受信息图

WebRTC通话

两个 peerConnection 的建立过程。首先客户端 A 和客户端 B 分别与信令服务器建立连接，连接建立之后客户端会分别创建 RTCPeerConnection 对象，将摄像头获取的本地 MediaStream 添加到 RTCPeerConnection 对象中，而后新建一个本地 offer，这个 offer 主要用于描述一些本机的网络和媒体信息，采用 SDP（Session Description Protocol）格式。通过 Websocket 服务将 offer 发送给目标连接方，目标连接方会创建与 offer 格式相同的 answer，用以告知请求连接方自身的信息。当两方任意一者都会通过调用 setRemoteDescription 来记录对方发送的 SDP 信息并让对方发送连接。当连接成功，本方会将 Add Stream 事件中的 event.stream 输入到 video 以得到实时影像，从而实现完整的 peerConnection 过程

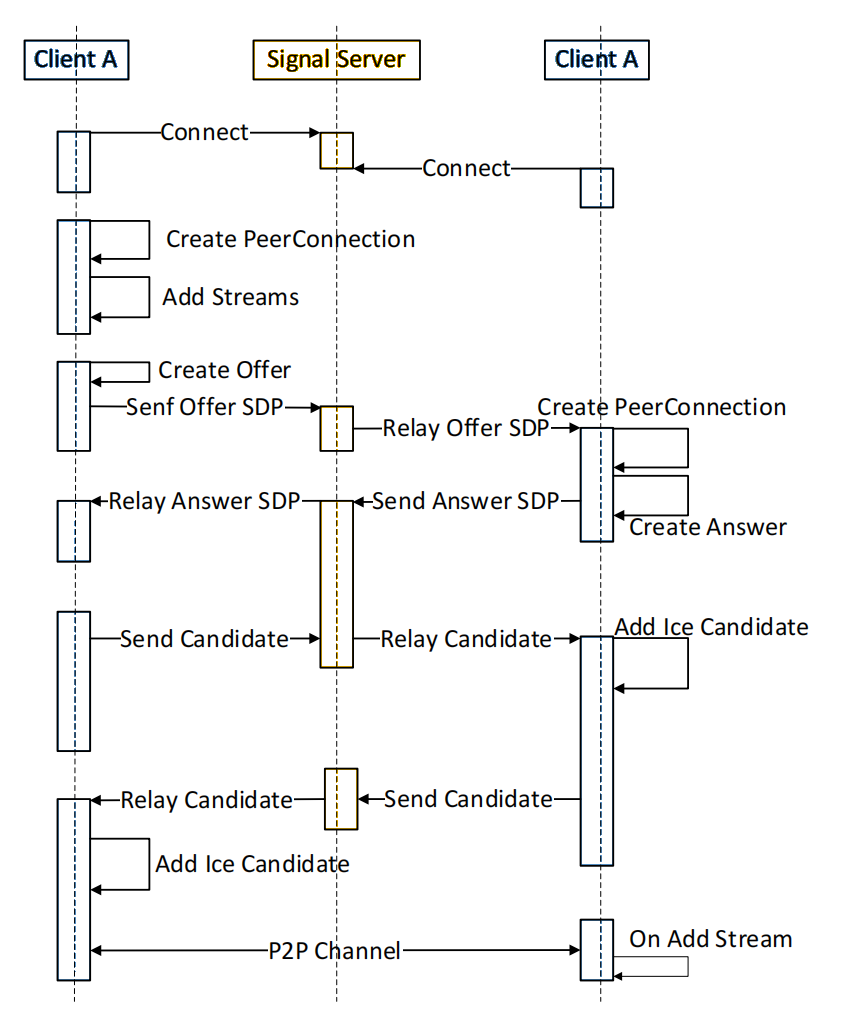


图4-3 前端通话流程图

建立到信令服务器的WebSocket连接，创建一个WebSocket实例对象，创建保存音频信息的对象，初始化ConstantObserver对象用于检测生成证书是否完毕，初始化ConstantObserver对象用于监听远程描述是不是已经完成，初始值为false，初始化ConstantObserver对象用于设置通话状态，值为HAS\_NO\_CALLING，代码如下：

let wsConnection = AppGlobal.wsConnection = new WSCo nnect ion("ws://loc alhost:3000/");

const callBtn = this.getElementById("call");

const receiveBtn = this.getElementById("receive");

const finishBtn = this.getElementById("finish");

const videophoneContainer = this.getElementById("videophone-container");

let currentAudio = {local: null, remote: null};

let iceServerConfig;

let generatingCertificate = new ConstantObserver(false);

AppGlobal.RTCConnection = new ConstantObserver(null);

AppGlobal.isCaller = new ConstantObserver(null);

AppGlobal.beforeAddTrack = new ConstantObserver(false);

let callingState = AppGlobal.callingState = new ConstantObserver(CallingS tate.HAS\_NO\_CALLING);

使用WebRTC内置的计算机密码学算法建立公钥私钥，配置信息如下：

* 算法名称: RSASSA-PKCS1-v1\_5
* hash：SHA-256
* 模长：2048
* 公共指数：5

代码如下：

RTCPeerConnection.generateCertificate({

name: "RSASSA-PKCS1-v1\_5",

hash: "SHA-256",

modulusLength: 2048,

publicExponent: new Uint8Array([1, 0, 1])

})

建立完毕后将检测生成证书是否完毕的ConstantObserver对象设置为已经初始化，在WebSocket连接建立成功后，为呼叫按钮注册一个点击事件回调函数，此函数代码如下：

let peerConnA = AppGlobal.RTCConnection.value = new RTCPeerCon nection(iceServerConfig);

let mediaStream = await navigator.mediaDevices.getUserMedia({audio: true/\*, video: true\*/});

for (let v of mediaStream.getTracks()) {

peerConnA.addTrack(v, mediaStream);

}

let offer = await peerConnA.createOffer()

await peerConnA.setLocalDescription(offer);

wsConnection.sendMessage(JSON.stringify(

new MessagePackage(offer, MessageType.DESCRIPTION)

))

监听RTCPeerConnection类的实例对象的icecandidate事件，如果此函数由icecandidate事件触发，实例化一个messageType为MessageType.CANDIDATE的MessagePackage的对象，并用JSON.stringify()方法打包成字符串发送至服务器：

if (e.candidate) {

wsConnection.sendMessage(JSON.stringify(

new MessagePackage(e.candidate, MessageType.CANDIDATE)

))

}

监听RTCPeerConnection类的实例对象的track事件，初始化Audio对象，设置媒体流为获取到的媒体流，自动播放，监听RTCPeerConnection类的实例对象的connectionstatechange事件，如果RTCPeerConnection类的实例对象的iceConnectionState变为connected，设置用于设置通话状态的ConstantObserver对象值为CallingState.CALLING，代码如下：

currentAudio.remote = new Audio();

currentAudio.remote.srcObject = e.streams[0];

currentAudio.remote.autoplay = true;

currentAudio.remote.muted = false;

AppGlobal.RTCConnection.value?.addEventListener("connectionstatechange", function () {

if (this.iceConnectionState === "connected") {

callingState.value = CallingState.CALLING

}

})

设置挂断按钮的点击事件回调函数，函数内容如下：关闭所有正在传输的媒体流并释放麦克风和扬声器资源，关闭当前的RTCPeerConnection，关闭通话页面，是否需要通知对等端关闭对话，如果需要的话通过发送一个CLOSE类型信息通知对等端通知对方同时挂断，设置用于设置通话状态的ConstantObserver对象值为CallingState.END，关闭通话页面，并且将所有已经传到监视器的事件全部清除，代码如下：

wsConnection.sendMessage(JSON.stringify(newMessagePackage(null, MessageType.CLOSE)));

remoteDescriptionIsReady.clearEventListener(ObserverCallBackType.change);

注册处理MessageType为MessageType.DESCRIPTION的MessagePackage的映射函数，创建WebRTC的RTCPeerConnection类的实例对象，打开通话页面，设置用于设置通话状态的ConstantObserver对象值为CallingState.REQUESTING，当接听者摁下接听键的时候，执行点击事件回调函数，代码如下：

letpeerConnA=AppGlobal.RTCConnection.value=new RTCPeerConnection(iceServerConfig);

receiveBtn.onclick = async function () {}

当接听者摁下接听键的时候，执行点击事件回调函数，流程为：创建一个Audio类的实例对象，获取用户麦克风设备，麦克风的流媒体赋给Audio类实例对象并设置自动播放，监听RTCPeerConnection类的实例对象的icecandidate事件，与上述icecandidate事件处理函数一致，监听RTCPeerConnection类的实例对象的track事件，添加麦克风的媒体流到待传输媒体流中，设置远程媒体描述为MessagePackage实例对象中的data属性字符串解析出的JSON对象，初始化应答并设置为本地媒体描述，实例化一个messageType为MessageType.REMOTE的MessagePackage的对象，并用JSON.stringify()方法打包成字符串发送至服务器，代码如下：

let mediaStream = await navigator.mediaDevices.getUserMedia({audio: true/\*, video: true\*/});

for (let v of mediaStream.getTracks()) {

peerConnA.addTrack(v, mediaStream);

}

await peerConnA.setRemoteDescription(messagePackage.data);

remoteDescriptionIsReady.value = true;

let answer = await peerConnA.createAnswer();

await peerConnA.setLocalDescription(answer);

wsConnection.sendMessage(JSON.stringify(

new MessagePackage(answer, MessageType.REMOTE)

));

注册处理MessageType为MessageType.CANDIDATE的MessagePackage的映射函数，此函数执行：注册处理用于监听远程描述是不是已经完成的ConstantObserver对象change事件的函数，将MessagePackage中的data属性中的JSON对象添加至RTCPeerConnection实例对象中的ICE候选，代码如下：

await AppGlobal.RTCConnection.value.addIceCandidate(messagePackage.data);

注册处理MessageType为MessageType.CLOSE的MessagePackage的映射函数，此函数执行：调用用户点击挂断的事件处理函数，将是否需要通知对等端设置为不需要，代码如下：

hungup(null, false);

注册处理MessageType为MessageType.REMOTE的MessagePackage的映射函数，此函数执行：将MessagePackage中的data属性中的JSON对象设置为RTCPeerConnection实例对象中的设置远程描述，代码如下：

await AppGlobal.RTCConnection.value.setRemoteDescription(messagePackage.data);

注册用于设置通话状态的ConstantObserver对象的change事件处理函数，当当前值为CallingState.CALLING的时候：触发callTimer模块中的start()方法开始计时。当当前值为CallingState.END，CallingState.HAS\_NO\_CALLING，CallingState.REQUESTING的时候：触发callTimer模块中的stop()方法开始停止计时，触发callTimer模块中的reset()方法开始重新计时，代码如下：

if (this.value === CallingState.CALLING) {

AppGlobal.callTimer.start();

} else if (

this.value === CallingState.END ||

this.value === CallingState.HAS\_NO\_CALLING ||

this.value === CallingState.REQUESTING

) {

AppGlobal.callTimer.reset();

AppGlobal.callTimer.stop();

}

传值映射以及状态变换的前端日志图：

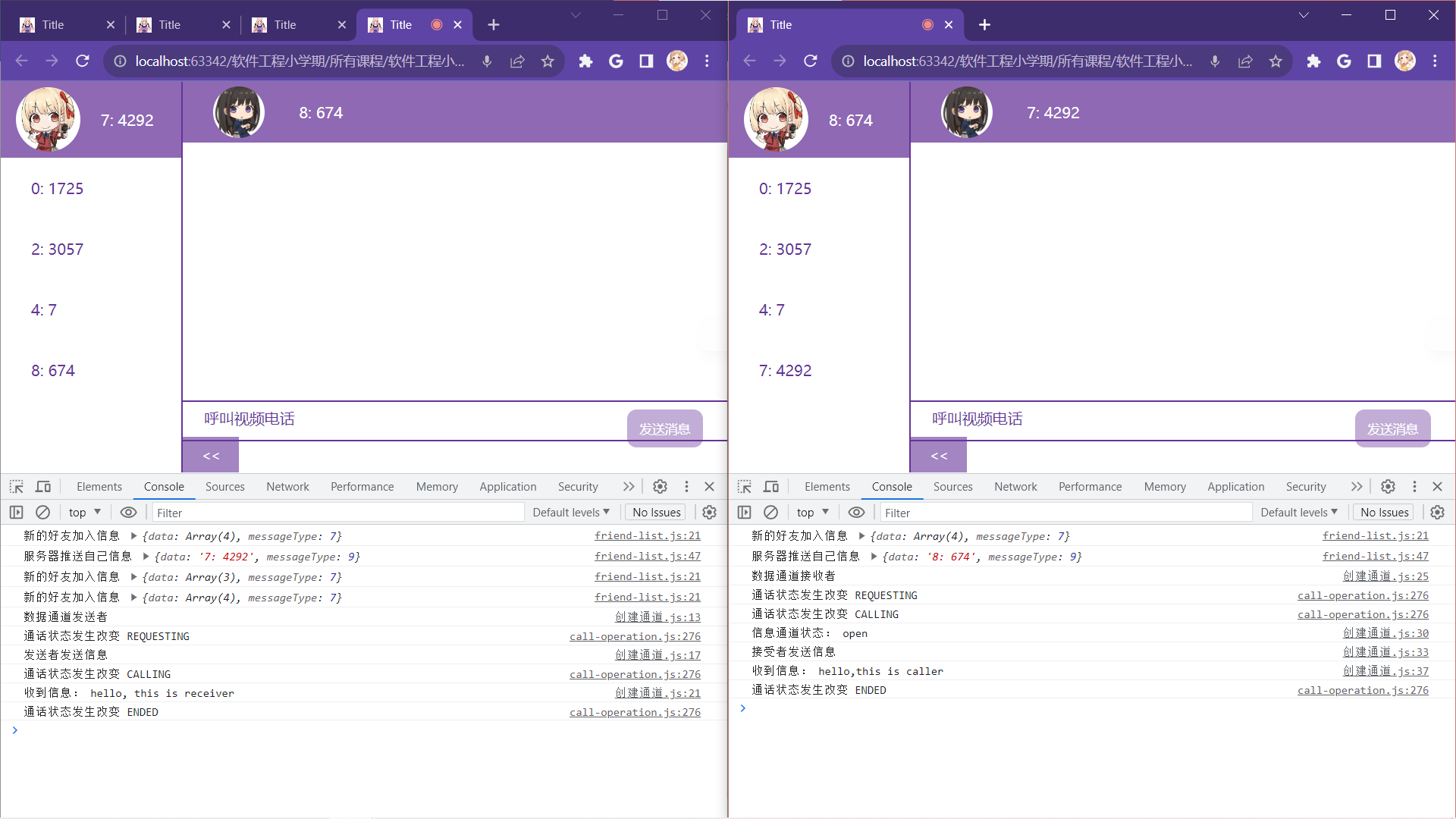


图4-4 前端状态改变图

### 4.2.3 信令服务器功能实现

信令服务器流程描述：

建立一个管理对象，用于存储HTTPS所需要的证书，证书通过文件读取获得，建立一个HTTP服务器，创建一个CallingManager，PeerManager类的实例对象，基于HTTP服务器建立WebSocket服务器，并注册回调函数，函数流程如下：

当某一端传过来字符串信息的时候，建立映射，当类型为ANOTHER\_ID时，处理代码如下：

con.twoPeer = new TwoPeers();

con.twoPeer.peer2 = peerManager.peers[messageObj.data];

con.twoPeer.peer1 = con;

当类型为MESSAGE时，处理代码如下：

if (!con.twoPeer) return;

let obj = JSON.parse(str);

obj.from = peerManager.peerToInfo.get(con).id;

obj.to = peerManager.peerToInfo.get(con.twoPeer.getAnotherPeer(con)).id;

SendMessage.sendMsg(con.twoPeer.getAnotherPeer(con), JSON.stringify(obj));

运行日志截图：

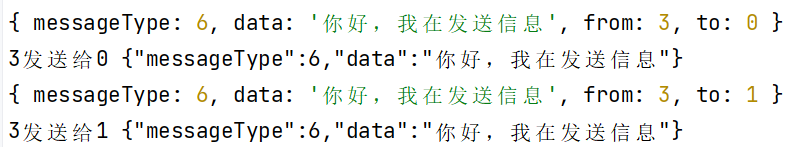


图4-5 发送信息日志图

当类型为CLOSE时，处理代码如下：

let peers = callingManager.getPeerAssociatedWith(con);

callingManager.removePeerAssociatedWith(peers[0].peer1);

callingManager.removePeerAssociatedWith(peers[0].peer2);

SendMessage.sendMsg(peers[0].getAnotherPeer(con), str)

其他时，处理代码如下：

if (!con.twoPeer) {

let peers = callingManager.getPeerAssociatedWith(con);

if (peers.length === 0) {

return;

} else if (peers.length > 1) {

return;

}

con.twoPeer = peers[0];

}

let peers = callingManager.getPeerAssociatedWith(con);

if (peers.length === 0) {

callingManager.addPeer(con.twoPeer);

peers = callingManager.getPeerAssociatedWith(con);

}

SendMessage.sendMsg(peers[0].getAnotherPeer(con), str);



图4-6 用户退出日志图

当某一端传过来关闭请求的时候：

peerManager.removePeer(this);

callingManager.removePeerAssociatedWith(this);

peerManager.peers.forEach(peer => {

let arr = structuredClone(allPeers);

arr.splice(arr.indexOf(peerManager.peerToInfo.get(peer).getInfo()), 1);

SendMessage.sendMsg(peer, JSON.stringify(

new MessagePackage(

arr,

MessageType.NEW\_FRIEND

)));

});

注册当新端加入的时候触发的回调函数，让WebSocket服务器监听3000端口，代码如下：

peerManager.addPeer(newPeer);

let allPeers = [];

peerManager.peers.forEach(peer => {

let arr = structuredClone(allPeers);

arr.splice(arr.indexOf(peerManager.peerToInfo.get(peer).getInfo()), 1);

SendMessage.sendMsg(peer, JSON.stringify(

new MessagePackage(

arr,

MessageType.NEW\_FRIEND

)));

})

SendMessage.sendMsg(newPeer, JSON.stringify(

new MessagePackage(

peerManager.peerToInfo.get(newPeer).getInfo(),

MessageType.SELF\_INFO

)));

});

wsserver.listen(3000);

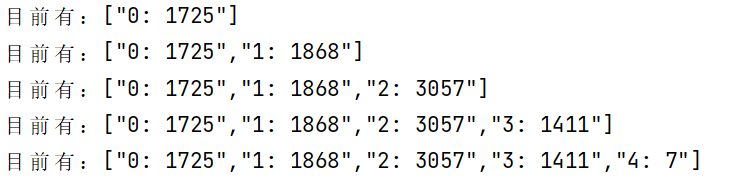


图4-7 添加用户日志图

### 4.2.4 发布页面服务器功能实现

**发布页面服务器流程描述：**

建立一个express APP，使用post请求body解析中间件，使用静态路径发表中间件，处理get请求路径为/index，回调函数为，设置推送时间，设置HTTP状态码为200，发送根目录下的p2p-calling.html页面，令Express APP监听4000端口。代码如下：

const express = require('express');

const bodyParser = require('body-parser');

const http = require('http');

const https = require('https');

const fs = require('fs');

const app = express();

const options = {

key: fs.readFileSync('../certificate/server.key'),

cert: fs.readFileSync('../certificate/server.crt')

};

http.createServer(options, app)

.listen(4000, () => console.log(`Server started on port ${4000}!`));

app.use(bodyParser.json());

app.use(express.static(\_\_dirname));

app.get('/index', (req, res) => {

res.status(200);

res.sendFile(root + "\\p2p-calling.html", options);

});

app.listen(4000, () => {

console.log('express server listened in 4000');

});

# 第5章 系统测试

软件系统测试是软件开发生命周期中非常重要的一环。它旨在验证和确认软件系统是否满足预期的需求和质量标准。通过系统测试，我们可以发现和修复潜在的缺陷、提高软件的可靠性和稳定性，并确保用户能够正常使用和满意地使用软件。在进行软件系统测试时，通常需要制定详细的测试计划和测试用例。测试计划定义了测试的范围、目标、资源和进度等信息，而测试用例则描述了具体的测试场景、输入数据和预期输出。通过执行测试用例，记录测试结果并分析缺陷，可以及时修复问题并确保软件系统的质量。

软件系统测试是确保软件质量和用户满意度的重要环节。通过功能测试、性能测试、安全测试和兼容性测试等多个方面的测试，我们可以发现和修复潜在问题，并持续提升软件系统的可靠性和性能。合理制定测试计划和执行测试用例，可以帮助我们全面评估软件系统的功能和性能，确保其符合用户需求和质量标准。

## 5.1 测试原则

软件测试的测试原则是指在进行软件测试过程中所遵循的基本准则和方法。这些原则旨在确保测试的有效性、可靠性和高效性，帮助测试人员更好地发现问题、提高软件质量。下面将介绍一些常见的软件测试的测试原则。

全面性原则：软件测试应该尽可能地覆盖所有可能的测试情况和路径，以发现潜在的问题和缺陷。全面性原则要求测试人员对所有功能、模块和边界条件进行测试，并考虑到各种正常和异常情况。通过全面性的测试，可以提高测试的可靠性和覆盖度，从而发现更多的问题并提高软件质量。

独立性原则：测试应该是独立于开发过程的，由专门的测试团队或测试人员完成。独立性原则有助于消除开发者的主观偏见，并确保测试具有客观性和公正性。测试人员应该与开发人员分开工作，独立地评估软件的功能和性能，并提供中立的测试结果和反馈。

及早测试原则：在软件开发周期的早期就开始进行测试是一个重要的原则。及早测试原则可以帮助在早期发现和解决问题，减少后期修复的成本和风险。通过在需求分析、设计和编码阶段进行测试，可以及时捕获和修复潜在的缺陷，并提高软件的可靠性。

缺陷跟踪原则：在软件测试过程中，需要建立一个有效的缺陷跟踪系统来记录和追踪缺陷。缺陷跟踪原则有助于收集、分类和优先处理缺陷，并确保及时跟进和解决这些问题。通过缺陷跟踪系统，可以提高沟通效率，加快缺陷修复的速度，从而提升整体的软件质量。

自动化原则：自动化测试是提高测试效率和准确性的重要手段。自动化原则要求在适当的情况下，利用合适的工具和技术，将重复和可自动化的测试任务交给机器完成。通过自动化测试，可以节省时间和人力成本，提高测试的覆盖度和一致性，并加速反馈和修复的过程。

风险导向原则：软件测试应该基于风险评估和优先级，重点关注对软件系统影响较大的功能和模块。风险导向原则要求测试人员根据软件的重要性、用户需求和可能的风险来确定测试策略和测试重点。通过集中资源和精力在高风险区域进行测试，可以更好地发现和解决潜在的问题。

可追溯性原则：测试用例和测试结果应该与需求和设计文档之间有明确的关联和追溯关系。可追溯性原则有助于跟踪测试的完整性和有效性，并帮助测试人员评估测试覆盖度。通过确保测试用例和结果与需求的一致性，可以提高测试的可靠性和准确性。

持续学习原则：软件测试是一个不断学习和改进的过程。持续学习原则鼓励测试人员不断学习新的测试方法、工具和技术，并将其应用到实际的测试工作中。通过不断学习和改进，测试人员可以提高自身的专业水平和技能，提高测试效率和质量。

适度原则：软件测试应该根据项目需求和时间约束进行适度的规模和深度。适度原则要求在保证测试质量的前提下，合理分配资源和时间进行测试。过度测试可能导致资源浪费和进度延误，而不足的测试可能无法发现潜在的问题。通过适度的测试，可以在满足项目需求的同时提高效率和成本控制。

持续交流原则：测试人员应与团队成员、开发人员和其他相关方保持持续的沟通和交流。持续交流原则有助于问题的及时解决和信息的共享，提高团队的协作效率和整体工作质量。通过积极参与讨论和反馈，测试人员可以更好地理解需求和设计，并提供有价值的测试建议和意见。

综上所述，软件测试的测试原则是指在进行软件测试过程中所遵循的基本准则和方法。这些原则包括全面性、独立性、及早测试、缺陷跟踪、自动化、风险导向、可追溯性、持续学习、适度和持续交流等。遵循这些原则可以帮助测试人员更好地发现问题、提高软件质量，并提升整体测试效率和效果。

## 5.2 测试用例

表5.1 系统兼容性测试用例

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 用例编号 | THXT-1-1 | 审核人 | | | | *盛钟锋* |
| 项目名称 | 基于WebRTC的通话系统 | 程序版本 | | | | 1.1.1 |
| 测试环境 | 硬件环境  服务器端：联想拯救者Y9000P  客户端：2台笔记本（CPU：Intel Core i5-11300，内存：16G 外存：1T） | | | | | |
| 软件环境  服务器端：操作系统——Windows11 ；Web服务器——Websphere 4.0  客户端：操作系统——Windows11；浏览器——Google chrome | | | | | |
| 网络环境  校园网，与服务器的连接速率为500M，与客户端的连接速率为500M | | | | | |
| 编辑人 | 门鑫博 | | 编辑时间 | | 2023年7月9日 | |
| 功能模块名 | 系统兼容性测试 | | | | | |
| 功能特性 | 可以在所有操作系统中运行 | | | | | |
| 测试目的 | 验证能否在不同操作系统中运行 | | | | | |
| 预置条件 | 程序无出错 | | | | | |
| 参考信息 | 需求说明中关于兼容性的信息 | | | 特殊规程说明 | | 无 |
| 用例编号 | 测试步骤 | 输入数据 | | 预测结果 | | 测试结果 |
| 1 | 在Windows11操作系统下运行进行通话调试 | 无 | | 可以正常运行 | | 可以正常运行 |
| 2 | 在Windows10操作系统下运行进行通话调试 | 无 | | 可以正常运行 | | 可以正常运行 |
| 3 | 在Windows7操作系统下运行进行通话调试 | 无 | | 可以正常运行 | | 可以正常运行 |
| 4 | 在Windowsxp操作系统下运行进行通话调试 | 无 | | 可以正常运行 | | 可以正常运行 |
| 5 | 在Linux操作系统下运行进行通话调试 | 无 | | 可以正常运行 | | 可以正常运行 |
| 6 | 在IOS操作系统下运行进行通话调试 | 无 | | 可以正常运行 | | 可以正常运行 |
| 7 | 在MacOS操作系统下运行进行通话调试 | 无 | | 可以正常运行 | | 可以正常运行 |
| 8 | 在HarmonyOS操作系统下运行进行通话调试 | 无 | | 可以正常运行 | | 可以正常运行 |

表5.2 信息发送功能测试用例

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 用例编号 | THXT-2-1 | 审核人 | | | 门鑫博 |
| 项目名称 | 基于WebRTC的通话系统 | 程序版本 | | | 1.1.1 |
| 测试环境 | 硬件环境  服务器端：联想拯救者Y9000P  客户端：2台笔记本（CPU：Intel Core i5-11300，内存：16G 外存：1T） | | | | |
| 软件环境  服务器端：操作系统——Windows11 ；Web服务器——Websphere 4.0  客户端：操作系统——Windows11；浏览器——Google chrome | | | | |
| 网络环境  校园网，与服务器的连接速率为500M，与客户端的连接速率为500M | | | | |
| 编辑人 | 郑瑞蓬 | 编辑时间 | | 2023年7月9日 | |
| 功能模块名 | 信息发送功能 | | | | |
| 功能特性 | 可以正确的发送信息 | | | | |
| 测试目的 | 验证输入不同类型的信息是否可以成功发送 | | | | |
| 预置条件 | 用户连接后选择好友进行对话 | | | | |
| 参考信息 | 需求说明中关于信息发送的信息 | | 特殊规程说明 | | 无 |
| 用例编号 | 测试步骤 | 输入数据 | 预测结果 | | 测试结果 |
| 1 | 点击好友后进行信息输入后发送 | 你好 | “你好”信息可以正常发送 | | “你好”信息可以正常发送 |
| 2 | 点击好友后进行信息输入后发送 | 46554654 | “46554654”信息可以正常发送 | | “46554654”信息可以正常发送 |
| 3 | 点击好友后进行信息输入后发送 | ！￥…… | “！￥……”信息可以正常发送 | | “！￥……”信息可以正常发送 |
| 4 | 点击好友后进行信息输入后发送 | “ ” | ““ ””信息可以正常发送 | | ““ ””信息可以正常发送 |
| 5 | 点击好友后进行信息输入后发送 | djkfrhksj | “djkfrhksj”信息可以正常发送 | | “djkfrhksj  ”信息可以正常发送 |
| 6 | 点击好友后进行信息输入后发送 | Dafafsdfssjhaskjhaskjhjghssddfasfasfasfasdsfasfasf65+4656154654等很多很多数据 | 发送出的消息不全 | | 发送出的消息不全 |

表5.3 视频聊天功能测试用例

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 用例编号 | THXT-2-3 | 审核人 | | | | 郑瑞蓬 |
| 项目名称 | 基于WebRTC的通话系统 | 程序版本 | | | | 1.1.1 |
| 测试环境 | 硬件环境  服务器端：联想拯救者Y9000P  客户端：2台笔记本（CPU：Intel Core i5-11300，内存：16G 外存：1T） | | | | | |
| 软件环境  服务器端：操作系统——Windows11 ；Web服务器——Websphere 4.0  客户端：操作系统——Windows11；浏览器——Google chrome | | | | | |
| 网络环境  校园网，与服务器的连接速率为500M，与客户端的连接速率为500M | | | | | |
| 编辑人 | 盛钟锋 | | 编辑时间 | | 2023年7月9日 | |
| 功能模块名 | 视频聊天功能 | | | | | |
| 功能特性 | 可以进行视频聊天 | | | | | |
| 测试目的 | 验证能够给不同的人进行视频聊天 | | | | | |
| 预置条件 | 用户连接后选择好友进行视频聊天 | | | | | |
| 参考信息 | 需求说明中关于视频聊天的信息 | | | 特殊规程说明 | | 无 |
| 用例编号 | 测试步骤 | 输入数据 | | 预测结果 | | 测试结果 |
| 1 | 点击第一位好友后进行视频聊天 | 无 | | 视频聊天正常 | | 视频聊天正常 |
| 2 | 点击第二位好友后进行视频聊天 | 无 | | 视频聊天正常 | | 视频聊天正常 |
| 3 | 点击第三位好友后进行视频聊天 | 无 | | 视频聊天正常 | | 视频聊天正常 |
| 4 | 点击第四位好友后进行视频聊天 | 无 | | 视频聊天正常 | | 视频聊天正常 |
| 5 | 点击第五位好友后进行视频聊天 | 无 | | 视频聊天正常 | | 视频聊天正常 |

## 5.3 缺陷分析

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 项目编号 | | THXT 1.1.1 | | 产品名称 | 基于WebRTC的通话系统 | |
| 委托单位 | | 第二组 | | 测试类型 | 黑盒测试 | |
| 测试时间 | | 2023/7/10 | | 测试地点 | 东北石油大学 | |
| 测试人员 | | 门鑫博 | | 日期 | 2023/7/10 | |
| 审核人员 | | 盛钟锋 | | 日期 | 2023/7/10 | |
| 问  题  汇  总 | 功能性测试 √ 严重（ ） 中等（ ） 微小（ ） 建议（ ）  可靠性测试 √ 严重（ ） 中等（ ） 微小（ ） 建议（ ）  性能测试 √ 严重（ ） 中等（ ） 微小（ ） 建议（ ）  用户文档 √ 严重（ ） 中等（ ） 微小（ ） 建议（ ） | | | | | |
| 序号 | 测试项 | | 子测试项  功能项 | 问题描述 | 缺陷类型 | 回归测试结果 |
| 1 | 兼容性测试 | | 操作系统兼容性测试 | 在进行操作系统兼容性测试时，发现无法在MacOS和iOS系统中无法正常通话 | 中等 | 通过 |
| 2 |  | |  |  |  |  |
| 3 |  | |  |  |  |  |
| 问题类型说明：  严重：Critical，Major  导致死机或误删信息或程序无法控制；  主要功能未实现或流程缺陷。  中等：Minor  一般功能点未实现或流程缺陷；  数据结果错误或界面信息错误。  微小：不影响功能实现的其他缺陷。Cosmetic  建议：建议的改进。 Other | | | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 项目编号 | | THXT 1.1.1 | | 产品名称 | 基于WebRTC的通话系统 | |
| 委托单位 | | 第二组 | | 测试类型 | 黑盒测试 | |
| 测试时间 | | 2023/7/10 | | 测试地点 | 东北石油大学 | |
| 测试人员 | | 郑瑞蓬 | | 日期 | 2023/7/10 | |
| 审核人员 | | 门鑫博 | | 日期 | 2023/7/10 | |
| 问  题  汇  总 | 功能性测试 √ 严重（ ） 中等（ ） 微小（ ） 建议（ ）  可靠性测试 √ 严重（ ） 中等（ ） 微小（ ） 建议（ ）  性能测试 √ 严重（ ） 中等（ ） 微小（ ） 建议（ ）  用户文档 √ 严重（ ） 中等（ ） 微小（ ） 建议（ ） | | | | | |
| 序号 | 测试项 | | 子测试项  功能项 | 问题描述 | 缺陷类型 | 回归测试结果 |
| 1 | 消息发送功能 | | 输入不同信息是否可以正常发送 | 当消息字数输入过多后，发送出的消息不全 | 中等 | 通过 |
| 问题类型说明：  严重：Critical，Major  导致死机或误删信息或程序无法控制；  主要功能未实现或流程缺陷。  中等：Minor  一般功能点未实现或流程缺陷；  数据结果错误或界面信息错误。  微小：不影响功能实现的其他缺陷。Cosmetic  建议：建议的改进。 Other | | | | | | |

## 5.4 测试总结

我们在此次测试中可以及时发现和修复系统中的问题和缺陷，提高系统的质量和可靠性。测试是一个持续的过程，需要不断优化和更新测试策略和方法，以确保项目成功交付符合用户期望的高质量通话系统。

功能测试：在功能测试中，我们验证系统是否按照需求规格书的要求正常运行。这包括呼叫建立、音视频传输、静音、通话保持等功能的测试。我们通过模拟各种使用情境和场景，检查系统在各种操作和交互中的表现，确保功能的正确性和一致性。

兼容性测试：WebRTC通话系统应支持多个平台和浏览器，因此兼容性测试是必不可少的。我们验证系统在不同操作系统（如Windows、Mac、Linux）以及不同浏览器（如Chrome、Firefox、Safari）上的兼容性。通过测试，我们可以确定系统是否在各种环境下都能正常运行，并提供一致的用户体验。

性能测试：性能测试旨在评估系统在高负载情况下的表现。我们测试呼叫连接时间、音视频传输质量、占用带宽等指标。通过模拟多个并发用户进行通话，我们可以评估系统的稳定性和性能，并确定其对高负载的承受能力。

安全性测试：安全性测试是保护用户隐私和数据安全的关键。我们对系统进行身份验证、数据加密等方面的测试，以确保用户通话内容和个人信息的安全。此外，我们还测试系统是否能有效地防御常见的网络攻击，如拒绝服务（DDoS）攻击。

可靠性测试：可靠性测试旨在评估系统的稳定性和可用性。我们测试长时间通话的稳定性，模拟网络断连后的恢复能力，并检查系统在各种异常情况下的表现。通过这些测试，我们可以确定系统在各种不可预测的情况下是否能够正常运行。

用户界面测试：用户界面测试着重于评估系统的用户友好性和易用性。我们测试界面布局、交互逻辑、多语言支持等方面，确保用户能够轻松地使用系统并获得良好的用户体验。通过定期收集用户反馈，我们可以不断改进和优化用户界面。

全局测试：在全局测试中，我们验证系统与其他相关组件（如电话网关、服务器）的集成和协同工作。我们确保系统在整个通信链路上的正常运行，并检查各个组件之间的数据传输和交互是否正确。

# 课程设计总结

课程设计简单描述。具体实现的功能如下：

1、音频通话：利用WebRTC可以在Web浏览器中进行实时的音频通话。用户可以通过浏览器直接与其他人进行通话，无需安装额外的插件或应用程序。；

2、视频通话：利用WebRTC可以在Web浏览器中进行实时的视频通话。用户可以通过浏览器直接与其他人进行通话，无需安装额外的插件或应用程序。；

3、文字交互：开发者可以使用DataChannel API在浏览器之间传输任意类型的数据，用户可以进行文字发送；

4、用户登录登出：用户可以在主页面进行登录登出操作；

5、联系人操作：可以进行联系人查询，增加、删除、修改联系人；

6、群聊：用户可以查询群聊，建立、解散、修改、退出群聊；

在系统开发过程中使用和扩展的关键技术如下：

前端开发技术：开发基于WebRTC的通话系统需要使用前端开发技术，如HTML、CSS和JavaScript。通过这些技术，可以构建用户界面、处理用户操作、管理媒体流等。

WebRTC API：WebRTC提供了一系列API，开发者可以利用这些API来实现音视频通话、数据传输、媒体流处理等功能。常用的WebRTC API包括getUserMedia、RTCPeerConnection、RTCDataChannel等。

信令协议：在WebRTC通话系统中，需要使用信令协议来进行连接的建立、媒体传输参数的交换等。常用的信令协议包括Session Initiation Protocol (SIP)、WebSocket、XMPP等。

服务器端技术：为了支持WebRTC通话系统的功能，通常需要搭建服务器端。服务器端技术可以使用Node.js、Java、Python等编程语言，利用相关框架来处理信令传递、房间管理、用户认证等。

STUN和TURN服务器：在WebRTC通话过程中，可能会遇到网络环境复杂的情况，例如存在NAT防火墙或无法直接连接的情况。为了解决这些问题，可以使用STUN（Session Traversal Utilities for NAT）服务器和TURN（Traversal Using Relays around NAT）服务器来提供中继和穿越功能。

音视频编解码：WebRTC通话系统需要对音频和视频进行编解码处理。常用的音视频编解码技术包括Opus、VP8、H.264等。通过使用这些编解码器，可以实现音视频的压缩和传输。

安全性技术：WebRTC通话系统涉及用户隐私和数据传输安全，因此需要考虑安全性技术。例如，使用HTTPS保护通信过程中的数据传输，使用SRTP（Secure Real-time Transport Protocol）对媒体流进行加密等。

系统存在的不足，具体体现在：

兼容性问题：尽管主流的浏览器都支持WebRTC，但不同浏览器之间仍然存在一些兼容性问题。特别是在移动设备上，由于硬件和软件的差异，可能会出现性能、稳定性以及媒体流处理方面的兼容性问题。

网络环境限制：尽管WebRTC可以穿越NAT并在复杂网络环境中工作，但某些网络配置下仍然存在限制。例如，存在防火墙或代理服务器的网络环境可能需要额外的配置才能确保正常的连接和通话。

服务器需求高：在建立和维护WebRTC通话过程中，需要使用STUN和TURN服务器来提供中继和穿越功能。这意味着在部署和维护通话系统时，需要考虑服务器资源的需求和成本。

媒体流控制和适应性：WebRTC通话系统对网络带宽和质量的要求较高。在网络延迟、抖动或丢包等情况下，通话的质量可能会下降。因此，开发者需要考虑如何进行媒体流控制和适应性调整，以提供更好的用户体验。

安全性问题：尽管WebRTC本身支持加密通信，但在实际应用中，可能会面临一些安全性问题。例如，可能存在跨站脚本攻击（XSS）或跨站请求伪造（CSRF）等攻击方式，开发者需要采取相应的安全措施来保护用户数据和隐私。

没有内置的录制和回放功能：WebRTC本身并没有提供内置的录制和回放功能，这对于某些应用场景可能是一个缺点。如果需要实现通话录制或回放功能，开发者需要额外考虑相关技术和工具的集成。

## 个人工作总结（门鑫博）

在这个软件工程项目中，我扮演了项目经理等多个角色，并负责了多个方面的工作。通过这个总结，我将回顾我在项目中所做的工作、遇到的挑战以及取得的成果。

项目背景和目标

我们的软件工程项目是为了开发一个基于WebRTC的通话系统。该系统旨在提供一个高效稳定且安全的p2p通话系统，通过网页面进行用户之间的视频语音或信息聊天等操作。项目的目标是实现一个稳定、高性能的通话系统，满足用户需求并提升客户满意度。

角色和职责

作为项目团队的一员，我的主要职责包括：

参与需求分析和功能设计

参与可行性分析研究

进行详细设计和总体设计

编写和维护代码库

进行单元测试和集成测试

协助解决技术问题和Bug修复

参与项目进展会议和沟通

工作内容和成果

我在项目期间完成了以下工作：

分析和理解用户需求，并与团队成员合作设计了核心功能和模块。

开发了前端界面，使用HTML、CSS和JavaScript等技术实现了用户信息交互和语音视频交互等效果。

使用JavaScript语言进行后端逻辑的实现。

进行了单元测试和集成测试，确保代码的质量和功能的正确性。

参与了团队的Bug修复工作，提高了代码的稳定性和可靠性。

作为项目经理和其他团队成员紧密合作，及时报告工作进展和遇到的问题，并提出解决方案。

遇到的挑战和解决方案在项目期间，我遇到了一些挑战，包括：

技术难题：在开发过程中，遇到了一些技术上的挑战，需要学习和研究相关技术，并与团队成员合作解决。通过查阅文档、参考示例代码和请教其他团队成员，我成功地克服了这些难题。

在进行项目开发的过程中我担任着项目经理，其中我进行了项目规划与管理：我制定了项目计划并确保项目按时交付。我负责定义项目目标、范围和任务，并将其分解为可管理的子项目和里程碑。我进行了风险评估和资源分配，并监控项目进度和预算。团队管理：我组建了一个高效的团队，并负责团队成员指导。我设定了明确的团队目标和角色职责，并建立了有效的沟通渠道和团队合作机制。需求分析与管理：我与客户紧密合作，收集和分析需求，并确保项目团队对需求有清晰的理解。我管理需求变更，并与团队协商解决潜在的需求冲突和优先级问题。资源协调与供应链管理：我与各方合作，确保项目所需的资源（人力、技术、设备等）的供应和协调。我与供应商合作，确保按时获取所需的技术支持和软硬件设备。

技术指导与问题解决：作为项目经理，我提供了对团队成员的技术指导和支持。我参与解决技术难题和风险，并确保系统的稳定性和性能。沟通与沟通管理：我负责与团队成员、客户和利益相关者进行有效的沟通。我组织了定期的会议和项目报告，确保各方对项目状态和进展有清晰的了解。风险管理与质量控制：我进行了风险评估和管理，制定并实施相应的风险应对策略。我确保项目团队遵循最佳实践和标准，以确保交付的产品质量符合预期。

在进行项目经理工作时使得我获得了很多的工作经验，比如项目管理和团队管理的实践经验，加强了我的沟通和协调能力，进行风险评估和问题解决的能力变强，同时对于web端开发的经验更加丰富，时间管理能力也得到了提升。

但是由于组内成员较少所以每个人都会身兼数职，其中我还担任了其他的角色，其中我还担任着程序员、前端工程师、后端工程师、测试员、发布工程师等角色。

其中在任职程序员时进行了如下工作：

在进行系统实现之前，由组内项目编程能力最强的人员进行代码块的分配，最后进行统计整合。

技术选型与架构设计：我参与了技术选型，并根据项目需求设计了系统的整体架构。我选择了适合的前端和后端技术，如HTML、CSS、JavaScript、Node.js等，以及相应的框架和库。

前端开发：我负责开发系统的前端界面，使用HTML、CSS和JavaScript等技术创建用户友好的交互式界面。我注重界面的美观性和用户体验，并确保界面的响应性和兼容性。

后端开发：我编写了系统的后端代码，使用Node.js和相关框架（如Express）来处理客户端请求，管理用户身份验证和会话管理，并与数据库进行交互。我遵循良好的编程规范和最佳实践，确保代码的可读性和可维护性。

实时通话功能实现：我使用WebRTC技术来实现系统的实时语音和视频通话功能。我处理了与媒体流的交互、网络连接的建立和维护，以及质量控制等方面的问题。我与团队成员协同工作，确保通话功能的稳定性和性能。

消息传递功能实现：我使用WebSocket协议来实现系统的实时消息传递功能。我编写了服务器端和客户端的代码，确保消息的可靠发送和接收，并实现了聊天室和个人对话等功能。

测试和调试：我进行了系统级别的测试和调试工作，包括单元测试、集成测试和性能测试。我修复了一些bug并优化了系统的稳定性和性能。

文档撰写：我记录了系统的设计和开发过程，并编写了用户文档和技术文档。这有助于团队成员之间的知识共享和项目的后续维护。

在担任程序员这个角色也使我对WebRTC技术的深入理解和应用能力，并且更加熟悉前端开发和后端开发的流程和技术栈，扎实的JavaScript、HTML和CSS编程能力。

在进行基于WebRTC的通话系统测试期间，我担任测试团队的一员，负责执行各项测试任务，并协助其他团队成员完成测试工作。以下是我在测试过程中所承担的职责和取得的成果：

1.功能测试：我负责验证系统能够成功建立音视频通话连接，并测试通话过程中的音频和视频质量。通过执行不同场景的测试用例，我发现并报告了一些功能缺陷，如无法正确接收远程视频等问题。

2.兼容性测试：在测试过程中，我积极尝试在不同浏览器和操作系统上进行测试，确保系统在各种平台上都能正常运行。

3.安全性测试：我参与了系统的安全测试，主要关注身份验证和数据加密方面的功能。

4.性能测试：我执行了一系列性能测试，模拟大量并发通话场景，评估系统在高负载下的稳定性和性能表现。通过性能测试，我成功发现了系统负载过高导致通话连接不稳定的问题，并提供了优化建议以改善系统的性能。

5.用户体验测试：我积极参与用户体验测试，邀请用户使用系统并提供反馈意见。根据用户的反馈，我与设计团队紧密合作，提出了一些建议和改进方案，以提升系统的易用性和用户满意度。

通过我们的努力和贡献，我们的测试团队成功发现并报告了多个问题，帮助开发团队及时修复了这些问题，并提升了系统的质量和稳定性。同时，我还与团队成员密切合作，分享测试经验和技术知识，促进了整个团队的合作与成长。我相信，我们的工作对于保证基于WebRTC的通话系统的质量和用户体验起到了重要的作用。

1. 时间分配

由于时间过于紧张，所以该项目有明确的时间表和里程碑，需要按时交付。为了应对时间压力，我进行了工作安排，优化代码开发和测试流程，并与团队成员紧密合作，确保项目进展顺利。

1. 学习与成长

通过参与这个软件工程项目，我获得了许多宝贵的经验和知识，包括：

* 深入理解软件开发流程和项目管理方法。
* 掌握了前端和后端开发技术，包括HTML、CSS、JavaScript等。
* 学会了与团队合作，有效沟通和协调工作。
* 提高了问题解决能力和学习能力，在面对挑战时保持积极的态度。

P2P协议学习：

在进行系统实现时，查阅了大量的相关资料并且学习总结了大量相关的知识，同事总结了大量其他同行大佬所编写的博客等；

P2P的应用从下载MP3被人们“认识”起，迅速风靡世界，目前运营上网络骨干流量的一半多都是P2P的流量，并且在文件共享、视频直播与点播、即时通(网上电话)、网络聊天、网络存储、网格计算等领域的应用发展神速。P2P的 组网模式与开发模式已经成为互联网上最适合“自由者社区”的网络模式，随着网络的普及，互联网已经走过信息上网、信息搜索的阶段，而“社区型”信息搜索可能成为下一阶段的标志。所以我们组打算利用现有的知识和曾经学过的知识进行基于WebRTC的通话系统

我个人理解的P2P发展的关键是P2P的业务模式，因为P2P真正体现了网状网络的优势，也从网络自身上解决了长期困扰人们的TCP/IP的服务保障问题，在“没有保障”的因特网上，实现了可以保障服务质量的应用，比如：SKYPE的通话、PPLive的视频直播。

其中有很多种不同类的P2P，其中我基于该项目大致的了解总结了三种分别是：集中式P2P网络、分布式P2P网络、混合式P2P网络。

集中式P2P网络：集中式P2P网络是一种对等（peer-to-peer）网络结构，其中存在一个或多个中央服务器充当协调者和中介。在这种网络中，所有的节点（也称为对等方或客户端）都与中央服务器进行通信和交互。在集中式P2P网络中，中央服务器起到重要的作用，负责管理网络的连接、数据传输和资源分配。它维护着网络拓扑结构的信息，并协调节点之间的通信。节点必须通过中央服务器来发现其他节点、建立连接和交换数据。节点之间的通信往往是通过中央服务器作为中转点进行的。当一个节点想要与另一个节点进行通信时，它首先向中央服务器发送请求，中央服务器然后将请求转发给目标节点。数据的传输也经过中央服务器，由中央服务器转发给目标节点。

集中式P2P网络特点和优势：

1.管理和控制：中央服务器能够有效地管理和控制整个网络，包括节点的加入和退出、连接的建立和断开、数据的路由和分发等。这样可以提高网络的稳定性和可靠性。

2.资源分配：中央服务器可以根据需求来分配和调度网络中的资源，例如带宽、存储空间等。这可以在一定程度上优化资源的利用，提高系统的性能和效率。

3.安全性：由于中央服务器的存在，集中式P2P网络可以更容易地进行身份验证、访问控制和安全策略的实施。这有助于保护网络和节点的安全。

分布式P2P网络：分布式P2P网络是一种对等（peer-to-peer）网络结构，其中所有的节点（也称为对等方或客户端）都具有相同的地位和功能，彼此之间直接通信和交互，而无需依赖中央服务器进行协调。在分布式P2P网络中，所有的节点都可以充当服务提供者和服务请求者。每个节点都能够直接与其他节点建立连接，并进行数据传输和资源共享。

分布式P2P网络优势特点：

1.去中心化：分布式P2P网络没有中央服务器的概念，所有的节点都具有平等的地位和权力，彼此之间进行直接的通信和交互。这使得网络更具自组织性和去中心化特点。

2.节点自治：每个节点都可以独立地做出决策，选择与哪些节点建立连接、共享哪些资源，并自主管理自己的数据和服务。这增加了网络的灵活性和可扩展性。

3.弹性和容错性：由于分布式P2P网络中的节点之间直接通信，即使某些节点离线或故障，其他节点仍然可以继续工作，从而提高了系统的容错性和稳定性。

4.资源共享：每个节点都可以共享自己的资源，例如处理能力、存储空间、带宽等。这使得节点之间可以相互协作，共同利用和分发资源，提高整个网络的效率和性能。

匿名性和隐私保护：在分布式P2P网络中，节点之间直接通信，没有中央服务器作为中介，可以更好地维护用户的匿名性和隐私保护。

混合式P2P网络：混合式P2P网络是一种结合了集中式和分布式两种网络结构特点的混合形式。在混合式P2P网络中，既存在中央服务器作为中介，又允许对等节点之间直接通信。

混合式P2P网络采用两层结构：上层是集中式的架构，下层是分布式的架构。中央服务器在上层起到协调和管理的作用，而对等节点则在下层进行直接的通信和数据传输。

在混合式P2P网络中，中央服务器负责维护对等节点的信息和网络拓扑，处理节点的加入和退出，以及路由和资源分配等任务。节点通过向中央服务器注册自己的信息，并从中央服务器获取其他节点的信息来建立连接。

一旦节点建立了连接，它们可以直接进行点对点的通信和数据传输，无需再通过中央服务器中转。这样可以减轻中央服务器的负载，提高系统的性能和可扩展性。

混合式P2P网络优势和特点：

1.灵活性：混合式P2P网络既兼顾了集中式架构的管理和控制特点，又拥有分布式架构的节点自治和资源共享特点。这使得网络在管理和性能方面更加灵活。

2.可扩展性：通过允许节点之间直接通信，混合式P2P网络可以更好地处理大规模网络的扩展性问题，减少中央服务器的负载压力，并提高整个网络的可扩展性。

3.中心化优势：中央服务器在混合式P2P网络中仍然发挥重要作用，可以提供更高级别的服务，如身份验证、资源索引和搜索等。这增加了网络的安全性和功能性。

4.数据一致性和可靠性：中央服务器可以帮助维护网络的数据一致性，并提供冗余备份和故障恢复机制，保证数据的可靠性和稳定性。

WebRTC学习总结

同时由于项目使用的框架是WebRTC，顾我们在技术员的带领下学习了WebRTC，其中WebRTC是一种开放标准和技术框架，用于在网页浏览器之间实现实时通信（RTC）。它允许开发人员通过简单的JavaScript API在浏览器中实现音频、视频和数据传输。

WebRTC基于P2P（点对点）通信原理，使用了一系列的协议和技术，包括实时传输协议（RTP）、会话描述协议（SDP）、网络穿越协议（ICE）等。它提供了强大的多媒体能力，能够实现语音通话、视频聊天、文件共享、屏幕共享等功能。同时使用WebRTC，用户无需安装任何插件或额外的软件，只需通过支持WebRTC的浏览器即可进行实时通信。目前，主流的浏览器如Google Chrome、Mozilla Firefox、Microsoft Edge等都已经原生支持WebRTC技术。

WebRTC在实时通信领域具有广泛的应用，包括在线会议、远程教育、客户服务、社交媒体、游戏等。它为开发人员提供了丰富的API，可以轻松地集成到现有的网站和应用程序中，为用户提供更加便捷和丰富的实时通信体验。

实现过程：实现信令服务器可自由选择（比如，socket.io、websocket等）。首先发起方获取视频流，如果成功，则新建「RTCPeerConnection对象」，然后创建offer，并发送给应答方。 addStream方法将getUserMedia方法中获取的流(stream)添加到RTCPeerConnection对象中，以进行传输onaddStream事件用来监听通道中新加入的流，通过e.stream获取 onicecandidate事件用来寻找合适的ICE，createOffer()是RTCPeerConnection对象自带的方法，用来创建offer，创建成功后调用setLocalDescription方法将localDescription设置为offer，localDescription即为我们需要发送给应答方的sdpsendOffer和sendCandidate方法是自定义方法，用来将数据发送给服务器，接收方也需要新建一个RTCPeerConnection对象，当应答方收到发起方发送的offer之后，调用setRemoteDescription设置RTCPeerConnection对象的remoteDescription属性，设置成功之后调用createAnswer方法，创建answer成功之后将其设置为localDescription，然后把answer发送给服务器，当发起方收到应答方发送的answer之后，将其设置为remoteDescription，至此WebRTC连接完成。此时虽然WebRTC连接已经完成，但是通信双方还不能直接通信，因为发送的ICE还没有处理，通信双方还没有确定最优的连接方式。应答方收到发起方发送的ICE数据时，调用RTCPeerConnection对象的addIceCandidate方法。remoteConnection.addIceCandidate(new RTCIceCandidate(ice))复制代码发起方收到应答方发送的ICE数据时，同样调用RTCPeerConnection对象的addIceCandidate方法。

WebRTC优势：

1. 方便。对于用户来说，在WebRTC出现之前想要进行实时通信就需要安装插件和客户端，但是对于很多用户来说，插件的下载、软件的安装和更新这些操作是复杂而且容易出现问题的，现在WebRTC技术内置于浏览器中，用户不需要使用任何插件或者软件就能通过浏览器来实现实时通信。对于开发者来说，在Google将WebRTC开源之前，浏览器之间实现通信的技术是掌握在大企业手中，这项技术的开发是一个很困难的任务，现在开发者使用简单的HTML标签和JavaScript API就能够实现Web音/视频通信的功能。

2. 免费。虽然WebRTC技术已经较为成熟，其集成了最佳的音/视频引擎，十分先进的codec，但是Google对于这些技术不收取任何费用。

3. 强大的打洞能力。WebRTC技术包含了使用STUN、ICE、TURN、RTP-over-TCP的关键NAT和防火墙穿透技术，并支持代理。

## 个人工作总结（郑瑞蓬）

很高兴能够在这个项目中担任前端开发的角色。在这个项目中，我主要负责前端部分的开发，包括编写HTML、CSS、JavaScript代码、调试代码、提高代码质量、优化用户体验等。在这个过程中，我学习了很多前端技术和知识，并且不断提高自己的技能水平。

### 软件工程学习工作总结

我在团队中主要担任设计师、程序员、测试员、架构师、前端工程师、后端工程师和运维工程师的工作任务

设计师负责软件用户界面和用户体验的设计。我在设计软件的时候主要工作包括：

需求分析：与客户和软件开发团队合作，收集和分析用户需求，并根据需求设计用户界面和用户体验。

原型设计：使用工具如Sketch、Figma、Adobe XD等制作原型，以便在开发之前验证和测试设计。

用户测试：与测试团队合作，进行用户测试以确保用户界面和用户体验符合用户需求和预期。

设计师在软件开发生命周期的早期阶段发挥重要作用，他们的设计决策对后续阶段的开发和测试具有重大影响。他们的工作成果决定了软件产品的视觉效果和用户体验，因此在软件开发中起着至关重要的作用。

程序员负责根据需求和设计规范编写和测试软件代码。我身兼程序员时主要干了以下工作。

编写代码：根据需求和设计规范编写高质量的、可读性良好的、可维护的软件代码。

测试代码：使用测试框架如Junit、TestNG等进行单元测试，确保代码的正确性和稳定性。

代码审查：与团队成员合作进行代码审查，以确保代码的质量和一致性。

程序员在软件开发生命周期的中间阶段发挥重要作用，他们的代码决定了软件产品的功能和性能。他们需要与其他团队成员密切合作，包括设计师、测试员、架构师和其他工程师，以确保软件产品的质量和可靠性。

测试员负责验证软件产品的质量和稳定性。我身兼测试员时主要干了以下工作。

编写测试计划：与团队成员合作，编写测试计划和测试用例，以确保软件产品的功能和性能符合用户需求和预期。

执行测试：执行各种测试，包括单元测试、集成测试、系统测试和验收测试等，以确保软件产品的质量和稳定性。

编写缺陷报告：记录和报告软件产品中发现的缺陷和问题，与团队成员合作确保问题得到解决。

测试员在软件开发生命周期的中后期发挥重要作用，他们的测试结果影响软件产品的质量和稳定性。他们需要与其他团队成员密切合作，包括设计师、程序员、架构师和其他工程师，以确保软件产品的质量和可靠性。

架构师负责设计和规划软件系统的结构和组件。我身兼架构师时主要干了以下工作。

设计系统架构：根据用户需求和业务需求设计系统架构，包括软件组件、数据流和系统接口等。

指导团队成员：与团队成员合作，提供技术指导和支持，确保开发符合系统架构和设计规范。

评估技术：评估新技术和工具，确定它们是否符合系统架构和业务需求，以及是否可以提高软件系统的性能和可靠性。

架构师在软件开发生命周期的早期阶段发挥重要作用，他们的决策对系统的结构和性能具有重大影响。他们需要与其他团队成员密切合作，包括设计师、程序员、测试员、前端工程师和后端工程师，以确保软件系统的可靠性和性能。

前端工程师负责实现和维护软件产品的前端界面。我身兼前端工程师时主要干了以下工作。

开发前端界面：根据设计规范和用户需求开发前端界面，包括HTML、CSS和JavaScript等。

测试前端界面：使用测试框架如Jest、Mocha等进行前端界面测试，以确保前端界面的正确性和稳定性。

与设计师合作：与设计师合作，确保前端界面符合设计规范和用户需求。

前端工程师在软件开发生命周期的中期发挥重要作用，他们的工作成果决定了软件产品的前端界面和用户体验。他们需要与其他团队成员密切合作，包括设计师、程序员、测试员、架构师和后端工程师，以确保软件产品的质量和可靠性。

后端工程师负责实现和维护软件产品的后端服务。我身兼后端程序员时主要干了以下工作。

开发后端服务：根据业务需求和系统架构开发后端服务，包括数据库设计和API开发等。

测试后端服务：使用测试框架如JUnit、TestNG等进行后端服务测试，以确保后端服务的正确性和稳定性。

与架构师合作：与架构师合作，确保后端服务符合系统架构和业务需求。

后端工程师在软件开发生命周期的中期发挥重要作用，他们的工作成果决定了软件产品的后端服务和性能。他们需要与其他团队成员密切合作，包括设计师、程序员、测试员、架构师和前端工程师，以确保软件产品的质量和可靠性。

运维工程师负责维护和管理软件系统的运行环境。我身兼运维工程师时主要干了以下工作。

部署和配置：部署和配置软件系统的运行环境，包括服务器、数据库和网络等。

监控和维护：监控软件系统的运行状态，及时发现和解决问题，以确保系统的稳定性和可靠性。

性能优化：优化系统性能，提高系统的响应速度和吞吐量等。

运维工程师在软件开发生命周期的后期发挥重要作用，他们的工作成果决定了软件产品的可靠性和稳定性。他们需要与其他团队成员密切合作，包括设计师、程序员、测试员、架构师和前后端工程师，以确保软件产品的质量和可靠性。

综上所述，设计师、程序员、测试员、架构师、前端工程师、后端工程师和运维工程师都在软件开发生命周期中发挥着重要作用。他们的工作内容和职责各不相同，但他们需要密切合作，相互支持，以确保软件开发过程的顺利进行和软件产品的质量和可靠性。在软件开发生命周期中，每个职位的贡献都是不可或缺的。

此外，在软件开发过程中，团队成员之间的沟通和协作也是至关重要的。团队成员应该定期进行会议和交流，以确保每个成员都了解项目的状态和进展。此外，使用协同工具如JIRA、Trello等可以帮助团队成员更好地管理和跟踪项目的进展。

最后，软件开发是一个不断迭代和改进的过程，团队成员应该持续关注用户反馈和市场变化，以及新技术和工具的发展。通过不断改进和升级软件产品，团队成员可以为用户提供更好的产品体验和价值。

软件开发是一个不断迭代和改进的过程，团队成员应该持续关注用户反馈和市场变化，并不断改进和升级软件产品，为用户提供更好的产品体验和价值。

### 技术学习工作总结

HTML5是HTML的第五次修订版本，它引入了许多新的元素，以提供更丰富的语义和功能。CSS3是CSS的最新版本，它提供了许多新的特性和功能，使得网页样式更加美观和灵活。JS（JavaScript）是一种轻量级的、解释型的或即时编译的编程语言，具有一等函数功能。它是最广为人知的网页脚本语言，但也被许多非浏览器环境使用，如Node.js, Apache CouchDB和Adobe Acrobat。我在本季度的学习目标是掌握JS的最新原生语法，了解它们的含义和用法，并在实际项目中应用它们。

我的学习目标是掌握新技术，了解它们的含义和用法，并在实际项目中应用它们。

我采用了以下几种学习方法：

阅读HTML5的官方文档和教程，如MDN，了解HTML5的新标签的定义、属性、示例和兼容性。

阅读CSS3的官方文档和教程，如MDN和BitDegree，参考了《CSS权威指南 第四版》（CSS: The Definite Guide 4th）文献，了解CSS3-CSS4的新属性的定义、属性、示例和兼容性。

阅读JS的官方文档和教程，如MDN和W3.ORG，了解JS的最新原生语法的定义、属性、示例和兼容性。

测试使用PHPStorm等工具，编写和运行代码，检查自己的理解和掌握程度。

参与线上线下的讨论和交流，与同学和其他开发者分享学习心得和经验，解决遇到的问题和困难。

通过本次的学习，我掌握了以下知识：

内容分区元素，如<article>、<aside>、<footer>、<header>、<main>、<nav>、<section>等，它们可以帮助我组织文档内容，提供更清晰的结构和语义。

表单元素，如<datalist>、<output>、<progress>、<meter>等，以及新的<input>类型，如color、date、email、number、range等，它们可以帮助我创建更丰富和友好的用户输入控件。

图形元素，如<canvas>、<svg>等，它们可以帮助我在网页上绘制动态的图形和图表。

媒体元素，如<audio>、<video>、<source>、<track>等，它们可以帮助我在网页上嵌入音频和视频内容，并提供控制和字幕功能。

交互元素，如<details>、<summary>、<dialog>等，它们可以帮助我创建可展开或隐藏的额外细节，或者弹出对话框或窗口。

我掌握了以下几类CSS3的新标签：

网格布局（Grid Layout），如grid、grid-template-columns、grid-template-rows、grid-gap等，它们可以帮助我创建基于网格的复杂和响应式的网页布局。

CSS变量（CSS Variables），如–var-name、var()等，它们可以帮助我定义和使用自定义的CSS属性值，方便修改和复用。

渐变（Gradients），如linear-gradient()、radial-gradient()等，它们可以帮助我创建平滑过渡的颜色效果，增加视觉吸引力。

动画（Animations），如@keyframes、animation、transition等，它们可以帮助我创建动态的交互效果，增加用户体验。

字体（Fonts），如@font-face、font-family、font-size等，它们可以帮助我使用更多的字体样式，增加文本表现力。

弹性盒布局（Flexbox Layout），如display: flex、flex-direction、flex-wrap、justify-content等，它们可以帮助我灵活地排列和对齐网页元素，适应不同的屏幕尺寸。

CSS函数（CSS Functions），如calc()、attr()、min()、max()等，它们可以帮助我进行一些计算或获取一些属性值，简化CSS编写。

我掌握了以下几类JS的最新原生语法：

CSS Houdini，是一组API，让开发者可以访问CSS引擎的底层部分，并使用它们来创建自定义样式和布局。例如，我可以使用CSS Houdini来创建自定义动画、变形、滤镜、画布等效果。

ES6模块化（ES6 Modules），是一种在JS中组织代码的方式，让开发者可以将代码分割成多个文件，并使用import和export语句来导入和导出变量、函数、类等。例如，我可以使用ES6模块化来创建可复用的组件、库、工具等。

WebSocket API（WebSocket API），是一种在浏览器和服务器之间建立双向通信的技术。例如，我可以使用WebSocket API来创建实时聊天、游戏、股票行情等应用。

EventTarget（事件目标），是一个接口，表示任何可以接收事件并具有事件监听器的对象。例如，我可以使用EventTarget来创建自定义事件，并在不同对象之间传递数据。

我掌握了以下几类WebRTC API：

RTCPeerConnection，是一个接口，表示一个连接到另一个对等端点的连接。我了解了如何使用它来创建或接受SDP，设置本地或远程描述，添加或移除ICE候选者和媒体轨道。

RTP扩展内容（RTP Extension Headers），是一种机制，允许在RTP包中添加额外的信息，以提供更多的功能或优化性能。我了解了如何使用它们来传递视频方向、音频级别、时间戳偏移等信息。

RTCRtpContributingSource（RTP贡献源），是一个接口，表示一个RTP流中的一个贡献源，即一个对媒体数据进行了修改或混合的源。我了解了如何使用它来获取贡献源的标识符、时间戳、音频级别等信息。

RTCRtpSynchronizationSource（RTP同步源），是一个接口，表示一个RTP流中的一个同步源，即一个产生媒体数据的源。我了解了如何使用它来获取同步源的标识符、时间戳、音频级别等信息。

RTCStatsReport（RTC统计报告），是一个接口，表示一个包含RTC相关统计信息的报告。我了解了如何使用它来获取RTCPeerConnection、MediaStreamTrack、RTCRtpSender、RTCRtpReceiver等对象的统计信息，如带宽、延迟、丢包、抖动等。

RTCDtlsTransport（DTLS传输），是一个接口，表示一个基于数据报传输层安全协议（DTLS）的传输层对象。我了解了如何使用它来获取传输层的状态、角色、指纹等信息，以及如何监听传输层的事件，如状态变化、错误等。

RTCDTMFSender（DTMF发送器），是一个接口，表示一个可以发送双音多频（DTMF）信号的对象。我了解了如何使用它来发送或停止发送DTMF信号，并获取当前发送的信号序列和持续时间等信息。

RTCIceTransport（ICE传输），是一个接口，表示一个基于交互式连接建立（ICE）协议的传输层对象。我了解了如何使用它来获取传输层的状态、角色、候选者等信息，以及如何监听传输层的事件，如状态变化、候选者收集、候选者对选择等。

RTCDataChannel（RTC数据通道），是一个接口，表示一个在RTCPeerConnection对象上建立的双向数据通道。我了解了如何使用它来创建或关闭数据通道，发送或接收数据，并获取数据通道的状态、标签、协议等信息。

通过学习，我收获了很多知识和技能，但也发现了自己的不足之处：

我对软件编程流程还不够熟练和灵活，有时候会忘记某些标签的属性或用法，或者混淆不同标签之间的区别和联系。

我对软件编程学习还不够深入和全面，有些标签我只了解了基本功能，没有探索更多的可能性和创意。

我对软件编程体系还不够兼容和优化，有些标签在不同的浏览器或设备上表现不一致或存在问题，需要进行更多的测试和调整。

## 个人工作总结（盛钟锋）

在本项目开发过程中我担任系统架构师等多个角色，通过本次项目开发既使我复习了以前学过的知识，又让我学会了很多新的知识，因为本次项目有完整的软件工程项目制作过程，因此我学会了很多软件工程过程方面的知识、例如：材料准备阶段的需求分析、概要设计、详细设计，在系统实现阶段学会了一些javascript代码的基础知识以及代码的整合，在系统测试阶段学会了很多做系统测试的方法。下面我将详细的介绍我学习到的知识。

软件工程学习：

软件工程过程是指在软件开发的各个阶段中，按照一定的流程和方法进行组织、管理和控制的过程。它包括需求分析、设计、编码、测试和维护等多个环节，其中每个环节都有相应的活动和任务。软件工程过程的实施对于开发高质量的软件产品具有重要意义。通过规范化的过程，可以提高软件开发的可控性、可预测性和可重复性，减少开发过程中的风险和错误，从而提高软件的质量和效率。常见的软件工程过程模型有：

瀑布模型：按照线性顺序依次完成需求分析、设计、编码、测试和维护等工作。

增量模型：将软件开发过程拆分为多个增量阶段，逐步完善软件功能。

迭代模型：通过多次迭代，不断完善软件功能和质量。

敏捷模型：强调快速响应需求变化、灵活适应的开发方式。

软件工程过程的关键活动有：

需求分析：对用户需求进行详细的调研和分析，明确软件功能和性能要求。

设计：根据需求分析的结果，设计软件架构和模块之间的关系，确定开发方案。

编码：根据设计文档，编写代码实现软件功能。

测试：对已编写的代码进行测试，包括单元测试、集成测试和系统测试等。

维护：对软件进行修复、更新和改进，保证软件的可用性和稳定性。

软件工程过程的管理与控制：

项目计划：制定合理的项目计划，明确任务和时间节点。

资源管理：合理分配人力资源和物资资源，确保开发过程的顺利进行。

风险管理：识别和评估项目中可能出现的风险，并采取相应的措施予以应对。

进度控制：及时监测项目进展情况，对延期或超出预算的情况进行调整和控制。

质量控制：通过严格的质量管理和测试，保证软件产品的质量达到用户要求。

软件工程过程的挑战和解决方案有：

需求变更：及时响应用户需求的变化，灵活调整开发计划和设计方案。

人力资源不足：合理规划和分配人力资源，进行团队协作，可以考虑外包或招聘额外的开发人员。

技术难题：及时关注技术发展趋势，引入新技术和工具，培训团队成员，提高技术水平。

进度控制困难：建立清晰的项目计划和进度表，定期检查和评估进度，及时调整和优化开发过程。

质量控制问题：加强测试环节，包括自动化测试、代码审查和质量评估等，确保软件质量符合标准。

软件工程过程是开发高质量软件产品的重要手段和方法。通过合理的规划、管理和控制，可以提高软件开发的效率和质量。同时，需要不断改进和优化软件工程过程，适应快速变化的需求和技术环境。

需求分析：

需求分析是软件开发过程中至关重要的一环，对于提高软件产品的质量和用户满意度起着决定性的作用。在我学习需求分析的过程中，我深入了解了需求分析的概念、方法和技巧，并通过实践项目来加深对其的理解和应用。以下是我对需求分析学习的总结：

一、学习目标与背景

在开始学习需求分析之前，我明确了自己的学习目标和背景。我希望能够全面了解需求分析的基本概念和原则，掌握需求分析的方法和技巧，以及熟练运用相关工具和模型进行需求分析。

二、基础知识的学习

为了掌握需求分析的基础知识，我系统地学习了需求工程的理论和方法。我阅读了相关的教材和论文，了解了需求分析的起源和发展历程，以及需求工程的主要概念和原则。我还学习了需求分析的各种方法和技术，如面谈法、问卷调查、用例建模等，以及常用的需求工具和模型，如用例图、活动图、状态图等。通过理论学习，我对需求分析的目的、过程和技术有了更清晰的认识。

三、实践项目的经验

为了将理论知识应用于实践，在这个项目中，我与团队成员和客户进行了频繁的沟通和讨论，收集了大量的用户需求和业务需求。我运用所学的方法和技巧对这些需求进行了深入分析和整理，并编写了详细的需求规格说明书。在需求评审和确认阶段，我与客户和团队成员合作，确保需求的准确性和一致性，并及时处理变更请求。通过参与实践项目，我深刻体会到了需求分析在软件开发过程中的重要性和挑战。

四、团队合作与沟通能力的培养

在需求分析的学习过程中，我意识到团队合作和良好的沟通能力对于需求分析的成功至关重要。在实践项目中，我与团队成员紧密合作，共同解决问题和把握需求。我学会了倾听和理解他人的观点，善于与他人沟通和协商，以达成共识和促进合作。通过团队合作，我不仅加深了对需求分析的理解，也提高了自己的团队合作能力。

五、问题与挑战的应对

在学习需求分析的过程中，我也遇到了一些问题和挑战。其中之一是如何处理需求变更。由于需求常常随着项目的推进而发生变化，我需要学会灵活应对需求的变更，并及时进行调整和修正。另一个问题是如何准确理解用户的需求。有时用户可能表达不清晰或矛盾的需求，我需要通过与用户进行深入的沟通和分析，帮助他们明确需求并找到最佳解决方案。我还遇到了时间和资源的限制，需要在有限的时间内完成需求分析工作，并合理分配资源来满足用户需求。

为了应对这些问题和挑战，我采取了以下措施：首先，我加强了与用户和团队成员的沟通和合作，积极倾听他们的意见和建议，并灵活调整需求分析的策略和方法。其次，我不断提升自己的专业知识和技能，通过阅读书籍和参加培训课程来拓宽视野和增强能力。最后，我学会了有效地管理时间和资源，制定合理的工作计划，并根据优先级来安排任务的执行顺序。

通过对需求分析的学习和实践，我对软件开发过程和用户需求有了更深入的理解和认识。我学会了运用各种需求分析方法和技巧，将用户的需求转化为明确的需求规格说明，为后续的设计和开发提供指导。我也意识到需求分析是一个复杂而关键的过程，需要团队合作、良好的沟通和协调能力，以及不断学习和提升自己的素质和技能。

在未来，我将在实践中不断积累经验和提高能力。我希望能够成为一名优秀的需求分析师，能够准确地理解和满足用户需求，为软件开发项目的成功交付做出贡献。同时，我也希望能够与团队成员和利益相关者密切合作，共同致力于实现高质量的软件产品和用户满意度的提升。

概要设计：

概要设计是软件开发过程中的重要一环，它可以帮助我们在开始编码之前更好地理解和规划整个项目。下面是我对概要设计学习的总结。

首先，我了解到概要设计是软件开发的第二个阶段，紧随需求分析之后。在需求分析阶段，我们通过与客户沟通和分析需求文档来确定系统的功能和特性。而在概要设计阶段，我们将这些需求转化为高层次的系统结构和模块设计。

其次，我明白了一个好的概要设计需要考虑多个方面。首先是系统的架构设计，即如何组织和管理各个模块以及它们之间的关系。良好的系统架构可以提高代码的可读性、可维护性和扩展性。其次是模块设计，即对每个模块进行详细的设计，包括模块的功能、接口和数据结构等。最后是用户界面设计，即如何设计一个直观、易用和美观的用户界面，以提供良好的用户体验。

进一步地，我了解到概要设计可以使用不同的工具和技术来支持。其中，UML（统一建模语言）是一个常用的工具，它提供了多种图形符号和术语，用于描述系统的结构和行为。通过使用UML，我们可以绘制用例图、类图、时序图等，以帮助我们更好地理解和交流系统设计。此外，面向对象设计原则也是概要设计中的重要参考依据，如开闭原则、单一职责原则、依赖倒置原则等。

在学习过程中，我积极参与了实践项目，并运用所学知识进行概要设计。通过实践，我发现概要设计能够帮助团队成员之间更好地协作和沟通，减少后期代码修改和调整的工作量。同时，良好的概要设计还有助于提高代码的质量和可靠性，减少错误的产生。

然而，我也遇到了一些挑战和困难。首先是对于较大规模的系统，如何划分模块和确定模块之间的关系。这需要考虑诸多因素，如功能耦合性、数据传递方式、模块复用性等。其次是如何平衡系统的灵活性和性能。有时候，为了提高系统的灵活性，我们需要引入更多的抽象和接口；而为了提高性能，我们可能需要冗余或优化代码。这都需要在设计中进行权衡。

综上所述，通过学习概要设计，我深刻认识到它在软件开发过程中的重要性。概要设计可以帮助我们更好地规划和组织整个项目，提高团队协作效率，减少后期修改的工作量，同时也有助于提高代码的质量和可靠性。尽管还有一些挑战和困难需要克服，但通过不断学习和实践，我相信我会越来越熟练和擅长概要设计。

详细设计学习总结：

详细设计是软件开发过程中非常关键的一环，它负责将概要设计转化为具体的代码实现。下面是我对详细设计学习的总结。

首先，我了解到详细设计是在概要设计之后进行的，它主要关注系统内部的模块、类和方法的设计。通过详细设计，我们可以更加具体地定义每个模块的功能、接口和数据结构等。同时，我们还可以确定具体的算法和数据结构来支持系统的实现。

其次，我明白了一个好的详细设计需要考虑多个方面。首先是模块的职责划分和接口设计。一个模块应该具有清晰的职责，并与其他模块通过良好定义的接口进行交互。良好的接口设计可以提高模块的可复用性和可测试性。其次是算法和数据结构的选择。根据需求和系统特点，我们需要选择合适的算法和数据结构来实现模块的功能。同时，我们还需要考虑算法的时间复杂度和空间复杂度，以确保系统的性能满足要求。最后是异常处理和错误处理的设计。我们需要考虑系统可能出现的异常情况，并设计合适的处理机制来保证系统的稳定性和可靠性。

进一步地，我了解到详细设计可以使用多种工具和技术来支持。其中，UML（统一建模语言）仍然是一个常用的工具。通过使用UML类图、时序图等，我们可以更加清晰地描述模块之间的关系和交互。此外，设计模式也是详细设计中的重要参考依据。设计模式提供了一些经过验证的解决方案，可以帮助我们解决常见的设计问题，提高系统的灵活性和可维护性。

在学习过程中，我积极参与了实践项目，并应用所学知识进行详细设计。通过实践，我发现详细设计可以帮助我们更好地理解和实现概要设计中定义的系统结构和功能。同时，良好的详细设计还有助于减少后期代码修改和调整的工作量，提高代码的可读性和可维护性。

然而，我也面临了一些挑战和困难。首先是对于复杂系统的详细设计，如何处理模块之间的复杂交互和依赖关系。这需要仔细分析和设计，确保每个模块的职责清晰，并明确接口的定义。其次是如何平衡详细设计的灵活性和性能。有时候，为了提高系统的灵活性，我们会引入更多的抽象和接口；而为了提高性能，我们可能需要优化代码或引入一些特定的数据结构。这都需要在设计中进行权衡。

综上所述，通过学习详细设计，我深刻认识到它在软件开发过程中的重要性。详细设计可以帮助我们将概要设计转化为具体的代码实现，提高系统的可读性、可维护性和性能。尽管还有一些挑战和困难需要克服，但通过不断学习和实践，我相信我会越来越熟练和擅长详细设计的技巧和方法。我将继续努力学习和实践，不断提升自己在详细设计方面的能力。

综上所述，在本次项目过程中我学习到了很多在以后工作过程中非常重要的知识，可以查缺补漏我的知识点，让我对软件项目开发有更加深刻的认识和理解，认识到一个成功的软件工程项目需要良好的团队合作和沟通。在项目中，与团队成员保持紧密的沟通和协作是至关重要的。其次，我了解到需求分析是项目成功的基石。第三，我意识到项目管理对于项目成功至关重要。另外，我了解到质量保证是一个高质量软件工程项目的关键要素。最后，我认识到持续学习和不断改进是软件工程项目中必不可少的因素。软件行业变化迅速，新技术和方法不断涌现。为了跟上行业的发展和变化，我们需要不断学习新知识、掌握新技术，并将其应用到实际项目中。同时，我们还需要反思和总结，从项目中吸取教训和经验，不断改进和完善自己的工作方法和流程。通过参与软件工程项目的学习和实践，我深刻认识到团队合作、需求分析、项目管理、质量保证和持续学习等方面对于项目成功的重要性。这些经验将成为我未来职业发展中宝贵的资产，我将继续努力学习和实践，不断提升自己在软件工程项目中的能力和水平。

# 参考文献

[1]刘映含. 基于WebRTC的实时视频图像增强技术研究[D].北京交通大学,2022:10.26-44.

[2]赵伯恺. 基于WebRTC的媒体服务器的设计与实现[D].北京邮电大学,2020:10.26-69.

[3]张灿. 基于WebRTC的实时图传系统的设计与实现[D].西安电子科技大学,2020:10.27-89.

[4]周华东.WebRTC实时通信开源技术介绍及实用案例[J].办公自动化,2020,25(04):14-16.

[5]袁永胜,赵魁元,肖峥瑜等.基于ICE框架的P2P技术在通讯领域的研究[J].价值工程,2022,41(23):145-147.

[6]鲍薇.P2P技术的分析与研究[J].电脑迷,2017(07):5.

[7]许至晶.JavaScript在Web开发的应用研究[J].信息记录材料,2022,23(12):242-244.

[8]向桂玲.JavaScript技术在Web网页中的应用研究[J].信息记录材料,2022,23(04):145-147.

[9]张博,于海洋.服务器端JavaScript技术分析[J].信息与电脑(理论版),2021(04):19-20.

[10]宋文瑞.基于JavaScript技术实现的面向对象编程方法[J].数码世界,2020(02):9-11.

[11]刘常宏.计算机软件工程管理与应用的策略分析[J].集成电路应用,2023,40(05):315-317

[12]刘健.软件工程技术在系统软件开发中的应用[J].互联网周刊,2022(20):38-40.

[13]韩迎红.软件工程中的Web开发技术应用[J].数字通信世界,2022(07):133-135.

[14]鲁洋.软件工程方法在计算机软件开发中应用[J].软件,2022,43(08):176-178.

[15]沈宏翔.软件工程方法在计算机软件开发中应用分析[J].电脑知识与技术,2022,18(07):59-60

# 附录1团队开发会议记录

## 团队会议记录1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 开发小组 | | 第二组 | 指导教师 | 王永安 |
| 团队组长 | | 门鑫博 | 记录人员 | 门鑫博 |
| 会议主题 | | 项目开发前准备工作 | 会议地点 |  |
| 会议时间 | | 2023-6-26 | 记录时间 | 2023-6-26 |
| 团队组员 | | 门鑫博、盛钟锋、郑瑞蓬 | 记录形式 | 电子文档 |
| 会议内容概要 | | | | |
| 1 | 可行性研究、人员分配和实施计划 | | | |
| 2 | 需求分析、任务概述和运行环境规定 | | | |
| 会议详情 | | | | |
| 1 | 项目可行性研究及人员分配实施计划：  在会议中进行了技术可行性和非技术可行性两方面，其中进行了技术选型，在组内的商讨下同意了使用JavaScript语言进行开发，同时我们在非技术可行性上主要对经济可行性、社会可行性、法律可行性、环境可行性等进行研究。  由于小组成员过少所以进行商讨后决定让每个成员进行多个职位工作，最后设定实施计划。 | | | |
| 2 | 需求分析、任务概述和运行环境规定：  （1）需求分析：会议商讨对功能进行规定和非功能规定，其中功能规划中分析了系统的功能需求进行了常规通话步骤的分解研究最后获得结果，并且在非功能规定中对响应速度、稳定性、安全性、可靠性等方面进行规划  （2）运行环境规定：规定了统一的运行环境如下：  处理器型号：Intel Core i5-7500 及以上  内存容量：8G以上  外存容量：256G以上 联机  输入输出设备：  摄像头：Logitech C920 HD Pro 联机  麦克风：Audio-Technica ATR2100x-USB 联机  扬声器/耳机：Bose QuietComfort 35 II 联机  显示器：ASUS ProArt PA278QV 联机  数据通信设备：  路由器：TP-Link Archer C7  网络适配器（网卡）：Realtek PCIe GbE Family Controller  功能键：无  其他专用硬件：无 | | | |
| 小组沟通情况总结 | | | | |
| 1 | 项目成员沟通顺畅，对于关键技术实现方法的交流让大家拓宽了自己程序开发的思路 | | | |
| 2 | 个人项目成员对讨论内容仍有疑问，经过详细沟通后，能够正视自己项目开发的问题，并积极进行改正 | | | |
| 纪律情况总结 | | | | |
| 1 | 团队无人迟到早退，按时参加会议 | | | |
| 2 | 团队成员服从领导和安排 | | | |

## 团队会议记录2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 开发小组 | | 第二组 | 指导教师 | 王永安 |
| 团队组长 | | 门鑫博 | 记录人员 | 郑瑞蓬 |
| 会议主题 | | 项目开发 | 会议地点 | 1D-504 |
| 会议时间 | | 2023-07-01 | 记录时间 | 2023-07-01 |
| 团队组员 | | 门鑫博、盛钟锋、郑瑞蓬 | 记录形式 | 电子文档 |
| 会议内容概要 | | | | |
| 1 | 项目开发技术选定 | | | |
| 2 | 系统基本功能模块设计 | | | |
| 3 | 系统开发进度安排 | | | |
| 会议详情 | | | | |
| 1 | 项目开发背景调研、平台熟悉：  WebRTC技术了解：首先，需要对WebRTC技术有一定的了解。WebRTC是一种实时通信技术，包括音视频传输、数据通信等功能。可以通过查阅官方文档、参考书籍、在线教程以及相关的技术博客和论坛来学习WebRTC的原理、API和使用方法。  相关项目案例研究：研究和分析已经实施过基于WebRTC的通话系统的项目案例。了解这些项目的设计、架构、功能、性能以及用户反馈等信息，从中学习经验和教训，并将其应用到自己的项目中。  平台选择：根据项目需求和技术要求，选择适合的WebRTC平台或框架。主流的WebRTC平台包括OpenTok、Twilio、Jitsi等，在选择时要考虑平台的稳定性、可扩展性、文档和社区支持等因素。  技术评估和验证：进行技术评估和验证，确保所选平台和工具能够满足项目需求。这可能包括搭建简单的原型、进行功能测试、性能测试等，以验证所选平台的适用性和可行性。  平台熟悉和实践：深入了解所选平台的API和功能，并进行实践。通过编写示例代码、参考文档和教程，熟悉平台提供的特性和工具，并进行一些简单的开发实验来加深对平台的理解和掌握。  调研市场和用户需求：进行市场调研，了解目标用户的需求和偏好，分析竞争对手的产品和服务。这有助于确定项目的定位和特色，从而在开发过程中更好地满足用户的期望和需求。 | | | |
| 2 | 系统基本功能模块设计：  （1）用户登录模块：完成验证用户身份并提供授权访问，实现了系统和用户之间的交互和沟通功能。  （2）联系人模块：完成用户之间的添加、删除、修改等功能。  （3）群聊模块：多个用户同时进行文字聊天、分享信息和进行协作。  （4）视频通话模块：双向视音频通信，使用户能够进行面对面的远程交流。  （5）文字交互模块：文字交互模块提供了实时的文字聊天功能，使用户能够方便地进行文字交流、分享信息和表达意见。 | | | |
| 3 | 系统开发进度安排：（共7天）  （1）框架搭建：2天  （2）系统架构设计：2天  （3）功能开发：3天 | | | |
| 小组沟通情况总结 | | | | |
| 1 | 项目成员沟通顺畅，充分了解彼此想法，有利于该项目的设计开发 | | | |
| 2 | 个人项目成员对讨论内容仍有疑问，经过详细沟通后，已明确思路，完成设计 | | | |
| 纪律情况总结 | | | | |
| 1 | 团队无人迟到早退，按时参加会议 | | | |
| 2 | 团队成员服从领导和安排 | | | |

## 团队会议记录3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 开发小组 | | 第二组 | 指导教师 | 王永安 |
| 团队组长 | | 门鑫博 | 记录人员 | 盛钟锋 |
| 会议主题 | | 项目总结工作 | 会议地点 | 1D-504 |
| 会议时间 | | 2023-07-12 | 记录时间 | 2023-07-12 |
| 团队组员 | | 门鑫博、郑瑞蓬、盛钟锋 | 记录形式 | 电子文档 |
| 会议内容概要 | | | | |
| 1 | 系统测试 | | | |
| 2 | 项目运营及维护 | | | |
| 会议详情 | | | | |
| 1 | 系统功能代码测试：  （1）在文字聊天过程中消息有轻微延迟，需要调整  （2）点击联系人时偶尔不加载聊天历史，需要调整  （3）视频通话过程中偶尔没有声音，需要调整  （4）视频通话过程中偶尔有接收延迟，需要调整  （5）文字聊天过程中偶尔接收不到消息，需要调整 | | | |
| 2 | 项目运营及维护：  1. 用户体验优化：关注用户的反馈和建议，以不断优化用户体验。这可以通过用户调研、用户测试和数据分析来了解用户需求并进行相应的改进。  2. 更新和升级：根据用户反馈和市场需求，及时进行项目的更新和升级。这可以包括修复漏洞、添加新功能、改进用户界面等。  3. 安全性和隐私保护：确保项目的安全性和用户隐私得到充分保护。这包括采取必要的安全措施，如加密通信、身份验证和防止恶意攻击。  4. 服务器和基础设施管理：确保项目所需的服务器和基础设施的稳定运行。这包括监控服务器性能、处理故障和漏洞修复等。  5. 市场推广：制定市场推广策略，扩大项目的知名度和用户群体。可以通过社交媒体宣传、广告投放、合作伙伴关系等方式来吸引新用户。  6. 数据分析和统计：收集和分析项目的数据，以了解用户行为、用户满意度和系统性能等方面的情况。这有助于做出更明智的决策和改进措施。  7. 持续监测和改进：定期监测项目的运行状态和用户反馈，及时发现问题并采取相应的改进措施。通过持续的监测和改进，确保项目始终保持良好的运行状态。  总之，项目的运营和维护需要综合考虑技术、用户需求和市场情况等因素。通过有效的管理和策略，可以确保项目的稳定性、安全性和用户体验，并不断提升项目的价值和竞争力。 | | | |
| 小组沟通情况总结 | | | | |
| 1 | 项目成员沟通顺畅，对于关键技术实现方法的交流让大家拓宽了自己程序开发的思路 | | | |
| 2 | 个人项目成员对讨论内容仍有疑问，经过详细沟通后，能够正视自己项目开发的问题，并积极进行改正 | | | |
| 纪律情况总结 | | | | |
| 1 | 团队无人迟到早退，按时参加会议 | | | |
| 2 | 团队成员服从领导和安排 | | | |

东北石油大学课程设计成绩评价表

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 课程名称 | | 软件工程课程设计 | | | | | | | | |
| 题目名称 | | 基于 JS WEBRTC的P2P通话系统 | | | | | | | | |
| 学生姓名 | | 门鑫博 | 学号 | 200702940824 | | 指导教师姓名 | 王永安 | | 职称 | 讲师 |
| 序号 | 评价项目 | | | | 评判标准 | | | | | 得分 |
| 1 | **系统分析：**能够依据软件工程原理进行可行性分析和需求分析。 | | | | 综合技术、法律、安全等多种因素完成可行性分析。 | | | 满分：10 | |  |
| 需求理解正确，需求建模正确、规范。 | | | 满分：10 | |  |
| 2 | **设计与开发：**能够采用结构化或者面向对象方法进行系统设计与开发。 | | | | 概要设计方案正确，详细设计合理正确，报告条理清晰、规范。 | | | 满分：10 | |  |
| 设计或者实现中具有一些独特的见解。 | | | 满分：10 | |  |
| 程序编写正确，功能丰富，系统测试充分。 | | | 满分：20 | |  |
| 3 | **项目管理：**能够制定合理的开发计划，并能正确地开展成本效益分析。 | | | | 课题进度计划合理，课题成本/效益分析正确、符合逻辑。 | | | 满分：15 | |  |
| 4 | **团队合作：**能够在软件工程实施中承担相应的角色。 | | | | 角色担当称职，团队分工合理，队员协作高效。 | | | 满分：15 | |  |
| 5 | **社会价值：**能够正确评价课题解决方案对社会的影响。 | | | | 课题开发目标明确，课题社会意义分析正确，阐述清晰。 | | | 满分：10 | |  |
| 总分 |  | | | | | | | | | |
| 教师评价：  （1）系统需求分析正确性、合理性和规范性： 优□良□中□及格差  （2）系统概要、总体设计的合理性和规范性： 优□良□中□及格差  （3）程序功能实现的正确性和测试的充分性： 优□良□中□及格差  （4）系统进度合理性和成本效益分析高效性： 优□良□中□及格差  （5）团队分工的合理性和成员协作的高效性： 优□良□中□及格差  （6）课题开发实用价值及对社会意义和影响： 优□良□中□及格差 | | | | | | | | | | |

指导教师： 2023 年 7 月 15 日

东北石油大学课程设计成绩评价表

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 课程名称 | | 软件工程课程设计 | | | | | | | | |
| 题目名称 | | 基于 JS WEBRTC的P2P通话系统 | | | | | | | | |
| 学生姓名 | | 盛钟锋 | 学号 | 200702940821 | | 指导教师姓名 | 王永安 | | 职称 | 讲师 |
| 序号 | 评价项目 | | | | 评判标准 | | | | | 得分 |
| 1 | **系统分析：**能够依据软件工程原理进行可行性分析和需求分析。 | | | | 综合技术、法律、安全等多种因素完成可行性分析。 | | | 满分：10 | |  |
| 需求理解正确，需求建模正确、规范。 | | | 满分：10 | |  |
| 2 | **设计与开发：**能够采用结构化或者面向对象方法进行系统设计与开发。 | | | | 概要设计方案正确，详细设计合理正确，报告条理清晰、规范。 | | | 满分：10 | |  |
| 设计或者实现中具有一些独特的见解。 | | | 满分：10 | |  |
| 程序编写正确，功能丰富，系统测试充分。 | | | 满分：20 | |  |
| 3 | **项目管理：**能够制定合理的开发计划，并能正确地开展成本效益分析。 | | | | 课题进度计划合理，课题成本/效益分析正确、符合逻辑。 | | | 满分：15 | |  |
| 4 | **团队合作：**能够在软件工程实施中承担相应的角色。 | | | | 角色担当称职，团队分工合理，队员协作高效。 | | | 满分：15 | |  |
| 5 | **社会价值：**能够正确评价课题解决方案对社会的影响。 | | | | 课题开发目标明确，课题社会意义分析正确，阐述清晰。 | | | 满分：10 | |  |
| 总分 |  | | | | | | | | | |
| 教师评价：  （1）系统需求分析正确性、合理性和规范性： 优□良□中□及格差  （2）系统概要、总体设计的合理性和规范性： 优□良□中□及格差  （3）程序功能实现的正确性和测试的充分性： 优□良□中□及格差  （4）系统进度合理性和成本效益分析高效性： 优□良□中□及格差  （5）团队分工的合理性和成员协作的高效性： 优□良□中□及格差  （6）课题开发实用价值及对社会意义和影响： 优□良□中□及格差 | | | | | | | | | | |

指导教师： 2023 年 7 月 15 日

东北石油大学课程设计成绩评价表

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 课程名称 | | 软件工程课程设计 | | | | | | | | |
| 题目名称 | | 基于 JS WEBRTC的P2P通话系统 | | | | | | | | |
| 学生姓名 | | 郑瑞蓬 | 学号 | 200702940816 | | 指导教师姓名 | 王永安 | | 职称 | 讲师 |
| 序号 | 评价项目 | | | | 评判标准 | | | | | 得分 |
| 1 | **系统分析：**能够依据软件工程原理进行可行性分析和需求分析。 | | | | 综合技术、法律、安全等多种因素完成可行性分析。 | | | 满分：10 | |  |
| 需求理解正确，需求建模正确、规范。 | | | 满分：10 | |  |
| 2 | **设计与开发：**能够采用结构化或者面向对象方法进行系统设计与开发。 | | | | 概要设计方案正确，详细设计合理正确，报告条理清晰、规范。 | | | 满分：10 | |  |
| 设计或者实现中具有一些独特的见解。 | | | 满分：10 | |  |
| 程序编写正确，功能丰富，系统测试充分。 | | | 满分：20 | |  |
| 3 | **项目管理：**能够制定合理的开发计划，并能正确地开展成本效益分析。 | | | | 课题进度计划合理，课题成本/效益分析正确、符合逻辑。 | | | 满分：15 | |  |
| 4 | **团队合作：**能够在软件工程实施中承担相应的角色。 | | | | 角色担当称职，团队分工合理，队员协作高效。 | | | 满分：15 | |  |
| 5 | **社会价值：**能够正确评价课题解决方案对社会的影响。 | | | | 课题开发目标明确，课题社会意义分析正确，阐述清晰。 | | | 满分：10 | |  |
| 总分 |  | | | | | | | | | |
| 教师评价：  （1）系统需求分析正确性、合理性和规范性： 优□良□中□及格差  （2）系统概要、总体设计的合理性和规范性： 优□良□中□及格差  （3）程序功能实现的正确性和测试的充分性： 优□良□中□及格差  （4）系统进度合理性和成本效益分析高效性： 优□良□中□及格差  （5）团队分工的合理性和成员协作的高效性： 优□良□中□及格差  （6）课题开发实用价值及对社会意义和影响： 优□良□中□及格差 | | | | | | | | | | |

指导教师： 2023 年 7 月 15 日