项目编号：

密 级：

JustChat详细设计

第23组

易嘉祯 201922080909

刘守池 201922080634

王思颖 201952080635

闵齐星 201922080637

牟泽培 201922081202

2019年12月13日

**变更记录**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 变更  编号 | 版本号 | 日期 | 章节/段落/行  或图/表号 | 变更  状态 | 变更简单描述 | 审核人 | 批准人 |
| 1 | v1 | 2019/11/29 |  | A |  |  |  |
| 2 | v2 | 2019/12/5 |  | M |  |  |  |
| 3 | V3 | 2019/12/13 |  | M |  |  |  |
| 4 |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |  |
| 10 |  |  |  |  |  |  |  |
| 11 |  |  |  |  |  |  |  |
| 12 |  |  |  |  |  |  |  |
| 13 |  |  |  |  |  |  |  |
| 14 |  |  |  |  |  |  |  |
| 15 |  |  |  |  |  |  |  |
| 16 |  |  |  |  |  |  |  |
| 17 |  |  |  |  |  |  |  |
| 18 |  |  |  |  |  |  |  |

变更状态：**A** – 增加 **M** – 修改 **D** – 删除

**目 录**

[1引言 1](#_Toc26473595)

[1.1编写目的 1](#_Toc26473596)

[1.2项目背景 1](#_Toc26473597)

[1.3术语 1](#_Toc26473598)

[1.4参考资料 1](#_Toc26473599)

[2任务概述 1](#_Toc26473600)

[2.1目标 1](#_Toc26473601)

[2.2开发与运行环境 1](#_Toc26473602)

[2.2.1开发环境的配置 1](#_Toc26473603)

[2.2.2运行环境的配置 2](#_Toc26473604)

[2.2.3测试环境的配置 2](#_Toc26473605)

[2.3需求概述 2](#_Toc26473606)

[2.4条件与限制 2](#_Toc26473607)

[3总体设计 2](#_Toc26473608)

[3.1软件描述 2](#_Toc26473609)

[3.2处理流程 2](#_Toc26473610)

[3.3功能结构设计 3](#_Toc26473611)

[3.3.1 通讯模块 3](#_Toc26473612)

[3.3.2 用户管理模块 3](#_Toc26473613)

[3.3.3 日志模块 3](#_Toc26473614)

[3.3.4 数据存取模块 4](#_Toc26473615)

[3.4功能需求与程序代码的关系 4](#_Toc26473616)

[5接口设计 4](#_Toc26473617)

[5.1外部接口 4](#_Toc26473618)

[5.2内部接口 4](#_Toc26473619)

[6数据结构设计 4](#_Toc26473620)

[6.1逻辑结构 4](#_Toc26473621)

[6.2物理结构（数据库表） 5](#_Toc26473622)

[6.2.1 用户表（User） 5](#_Toc26473623)

[6.2.2 组聊表（Group） 5](#_Toc26473624)

[6.2.3 话题表（Topic） 5](#_Toc26473625)

[6.2.4 消息表（Message） 5](#_Toc26473626)

[7界面设计 6](#_Toc26473627)

[7.1主界面 6](#_Toc26473628)

[7.2创建新话题 6](#_Toc26473629)

[7.3查看所有话题 6](#_Toc26473630)

[7.4查看某个具体话题 6](#_Toc26473631)

[7.5查看我的话题 7](#_Toc26473632)

[7.6创建新组聊 7](#_Toc26473633)

[7.7查看所有组聊 7](#_Toc26473634)

[7.8查看某个具体组聊 7](#_Toc26473635)

[7.9查看我的组聊 7](#_Toc26473636)

[9出错处理设计 7](#_Toc26473637)

[10安全保密设计 8](#_Toc26473638)

[11维护设计 8](#_Toc26473639)

# 1引言

## 1.1编写目的

即时消息系统是一种在后PC时代兴起的，以Internet网络为基础的，允许交互双方即时地传送文字、语音、视频等信息，能够跟踪网络用户在线状态的网络应用软件。本概要设计旨在对初期的软件进行一个框架设计，再自顶向下进行细化。确定软件范围，对所开发软件的功能、性能、用户界面及运行环境等做出详细说明，制定完整的信息描述、恰当的检验标准以及其他与需求有关的数据。

## 1.2项目背景

项目名称：JustChat

委托单位：

开发单位：

## 1.3术语

【列出本文档中所用到的专门术语的定义和外文缩写的原词组。】

|  |  |
| --- | --- |
| 缩写、术语 | 解 释 |
|  |  |
|  |  |

## 1.4参考资料

# 2任务概述

## 2.1目标

同一局域网内匿名聊天的桌面应用程序。主要分三块：

1、群聊，即同一局域网内所有人在同一个大群里进行聊天；

2、组聊，带有某一特定主题的小组内多人聊天；

3、意见征集，即用户可发布一个话题帖子征集意见，形如“如何看待XX”，其它用户可以发表自己的看法，软件会实时分析出综合意见（百分比形式的正面/负面）。

## 2.2开发与运行环境

### 2.2.1开发环境的配置

【说明本系统应当在什么样的环境下开发，有什么强制要求和建议。】

|  |  |  |
| --- | --- | --- |
| 类别 | 标准配置 | 最低配置 |
| 计算机硬件 | Memory:8G cpu:3.0GHz | Memory:1Gcpu:1GHz |
| 计算机软件 | vs2019，Qt5.8.0 | vs2015(以上)，Qt5.8.0 |
| 网络通信 | TCP/IP | TCP/IP |
| 其它 |  |  |

### 2.2.2运行环境的配置

|  |  |
| --- | --- |
| 类别 | 标准配置 |
| 计算机硬件 | Memory:1G cpu:1GHz |
| 软件 | vs2019 |
| 网络通信 | TCP/IP |
| 其它 |  |

### 2.2.3测试环境的配置

Windows平台，Win10

## 2.3需求概述

详见《软件需求规格说明书》

## 2.4条件与限制

暂无

# 3总体设计

## 3.1软件描述

本系统是基于P2P的局域网即时匿名通讯软件，帮助在局域网中有临时通讯需求的用户传递文本信息和征求意见。用户在某一局域网内上线进入软件即可开始聊天，个人信息仅有一个ID号，软件会一直保存个人痕迹在本地文件中，并在每次启动时询问是否需要清理之前的痕迹。

## 3.2处理流程

**上线过程**：

加载个人相关的痕迹（若存在）；

若不存在个人痕迹，则广播上线消息请求回应；

收到回应，设置属于自己的唯一ID（局域网内）；

显示主界面。

**下线过程**：

保存个人痕迹到本地文件；

广播离线消息；

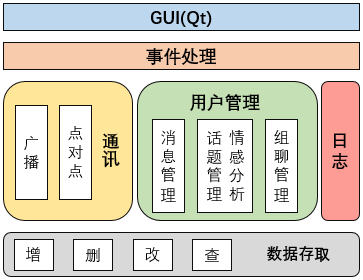
退出

**各类信息：**

1. 消息类：群聊消息、组聊消息、话题评论消息。
2. 用户操作类：创建组聊、发送消息、发送评论、创建话题，删除组聊，删除话题，查看话题，查看组聊。

## 3.3功能结构设计

软件整体架构如下图所示。



### 3.3.1 通讯模块

通讯模块主要处理用户之间的消息传输，主要分为广播通讯和组聊通讯。

1、广播：采用TCP/IP协议，处理五类消息：分别是：

（1）基本的群聊消息（针对局域网内在线的所有人）；

（2）新用户上线后广播请求获取其它人的ID以设置自己的唯一ID；

（3）用户建立/删除话题后通知所有人；

（4）用户建立/删除组聊后通知所有人；

（5）每个话题的评论消息以及实时分析的综合意见会广播给所有人。

2、点对点，同样采用TCP/IP协议。点对点主要处理一类消息，即每个组聊内部的消息，组聊内部的消息会点对点发到所有在当前组聊的用户那里去。

开启单独的通讯线程进行处理

### 3.3.2 用户管理模块

1、普通的消息管理。无论是群聊消息、组聊消息还是话题评论，都通过调用通讯线程进行处理，区别在于传输的数据头部不同（以区分不同消息），另外根据消息种类选择广播还是点对点。

2、话题管理（情感分析）。主要处理用户的话题创建操作、删除操作以及对话题评论的实时情感分析。

3、组聊管理。主要处理用户的组聊创建操作、删除操作。

### 3.3.3 日志模块

即记录软件运行过程中的事务流水，按日志级别(level: 0~4)来进行记录，数值越高，则表示故障越严重。

开启单独的日志线程进行处理。

### 3.3.4 数据存取模块

数据存取模块涉及各类消息的存取和日志文件的存取，其中各类消息采取Json文件方便存取，日志文件则直接作普通文本文件处理。

说明：不使用数据库，直接使用文件存取，所以还有自己实现的增删查改操作接口。

## 3.4功能需求与程序代码的关系

暂无。

# 4详细设计

# 4.1时序图

## 4.1.1广场群聊时序图

## 4.1.2创建新话题时序图

## 4.1.3查看所有话题时序图

## 4.1.4查看我的话题时序图

## 4.1.5创建新组聊时序图

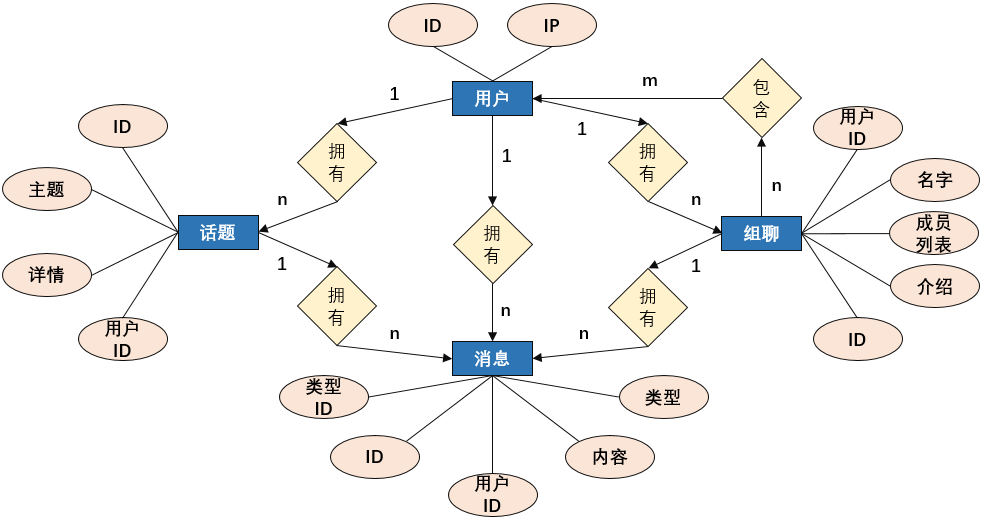
## 4.1.6查看已有组聊时序图

## 4.1.7查看我的组聊时序图

# 4.2数据结构设计

## 4.2.1逻辑结构

主要包含四个部分的数据，它们自身包含的属性以及相互之间的关系如下图所示：



## 4.2.2物理结构（数据库表）

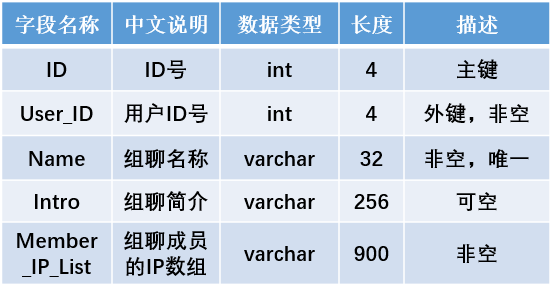
数据库定义

|  |  |
| --- | --- |
| **表名** | **描述** |
| User | 用户信息表，用于在某一局域网内唯一确定每个用户。 |
| Group | 组聊信息表，通过组聊创建者ID和组聊ID唯一确定组聊，保存组聊名称、简介信息，并记录组员的IP。 |
| Topic | 话题信息表，通过话题创建者ID和话题ID唯一确定话题，保存话题名称和详情信息，使用所有在线用户的IP。 |
| Message | 聊天消息信息表，通过用户ID确定二者之一（组聊、话题）下具体某一个组聊或话题的消息具体内容。 |

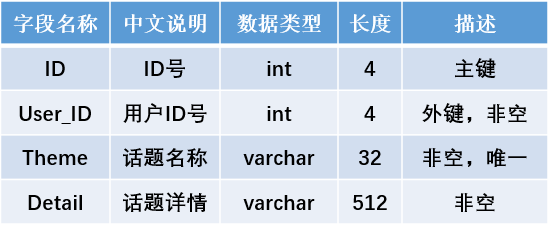
### 4.2.2.1 用户表（User）



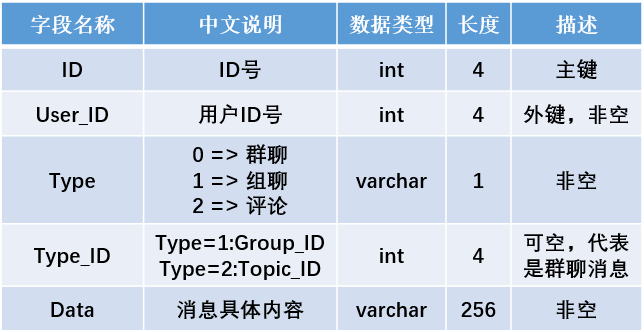
### 4.2.2.2 组聊表（Group）



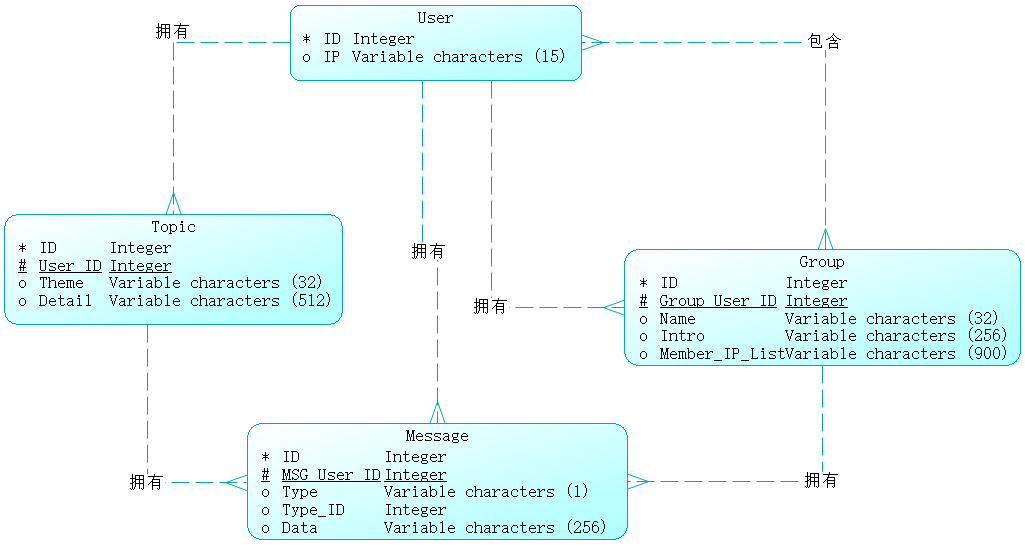
### 4.2.2.3 话题表（Topic）



### 4.2.2.4 消息表（Message）



## 4.2.3物理结构（数据视图）



# 4.3接口设计

## 4.3.1外部接口

用户通过输入设备（键盘、鼠标）、输出设备（显示器）与软件交互，软件通过与这些设备的接口捕获用户信息。通信采用TCP/IP。

### **4.3.1.1通讯模块**

主要用于底层数据包的传输，不考虑数据包的内容，提供UDP（P2P，广播）、TCP（点对点）的发送和接收功能。其中前者主要处理较短文本字符串的传输，后者主要用于文件的传输。

对外接口（供上层调用）：

1、/\*

功能说明：Udp点对点发送

传入参数：data - 需要传输的数据包、receiverIp - 传输目标Ip地址

\*/

void UdpSendP2P(const QByteArray &data, const QString &receiverIp);

2、/\*

功能说明：Udp局域网内广播发送

传入参数：data - 需要传输的数据包

\*/

void UdpSendBroadcast(const QByteArray &data);

3、/\*

功能说明：Tcp点对点发送

传入参数：data - 需要传输的数据包、receiverIp - 传输目标Ip地址

\*/

void TcpSendP2P(const QByteArray &data, const QString &receiverIp);

对外信号（连接后触发外部事件，Qt信号槽机制）：

1、/\*

功能说明：Udp有数据到来时发出相应信号

传出参数：data - 接收的数据包、senderIp - 传输来源Ip地址

\*/

void UdpReceive(QByteArray\* data, QString\* senderIp);

2、/\*

功能说明：Tcp有数据到来时发出相应信号

传出参数：data - 接收的数据包

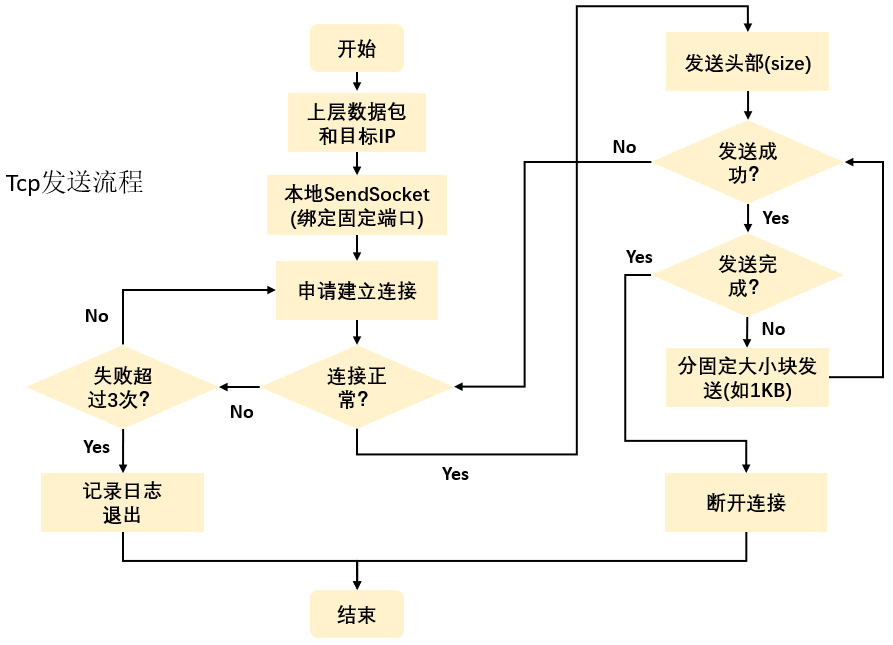
\*/

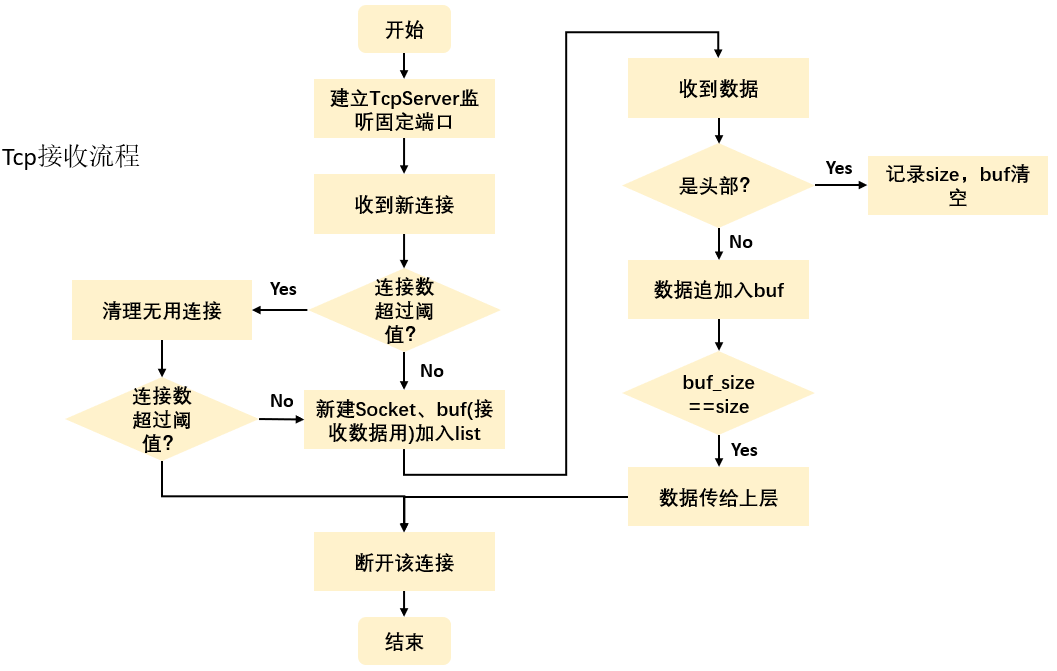
void TcpReceive(QByteArray\* data);

实施细节：

1、Udp通信，逻辑比较简单，分为专门处理接收的套接字（SendSocket）和专门处理发送的套接字（RecvSocket），分别绑定不同的固定端口号，当有相应事件来临时直接调用

2、Tcp通信





### **4.3.1.2日志模块**

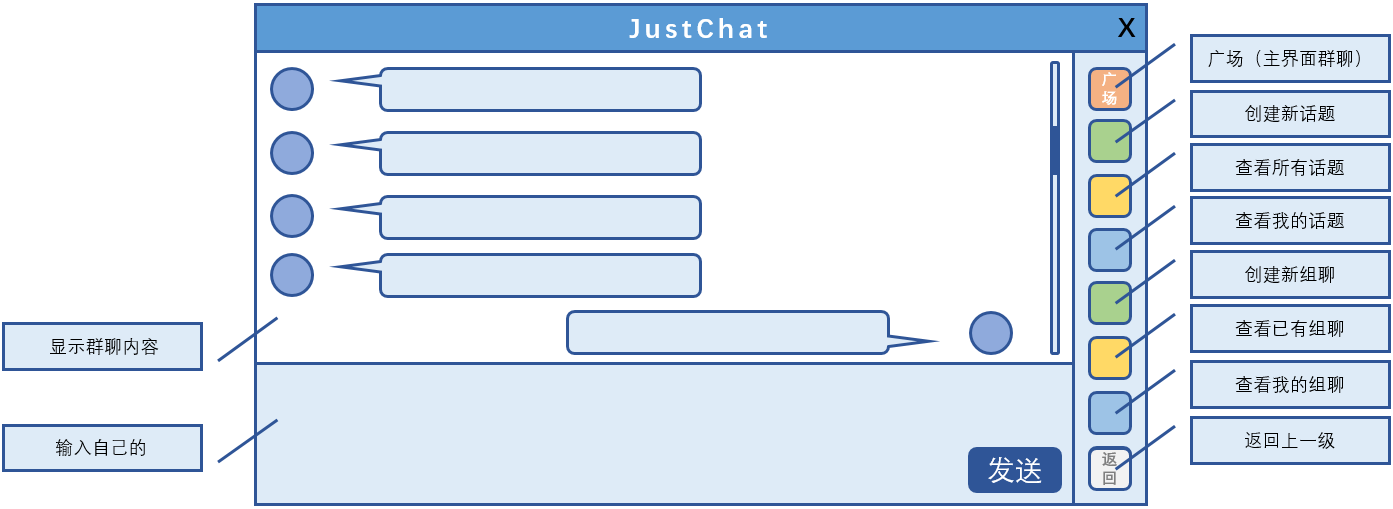
## 4.3.2内部接口

【说明该软件各个模块之间接口关系。】

# 5界面设计

## 5.1主界面

主界面是用户进入软件后看到的第一个界面，即群聊，又名广场。



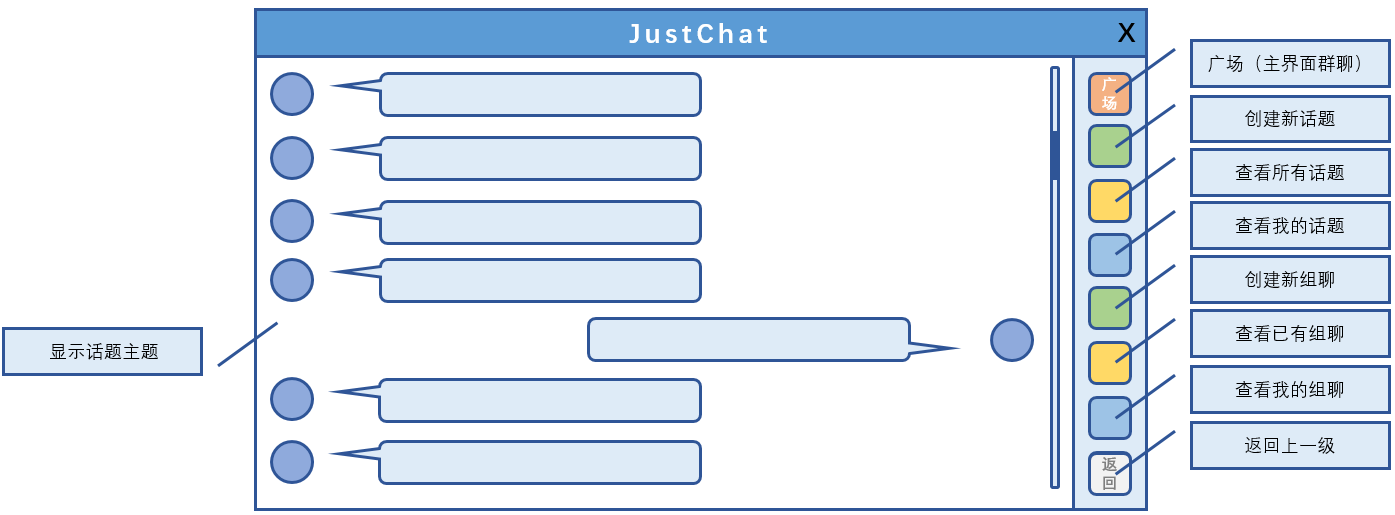
## 5.2创建新话题

创建新话题是弹窗形式，创建成功会居中弹框提示，然后进入话题详情页。



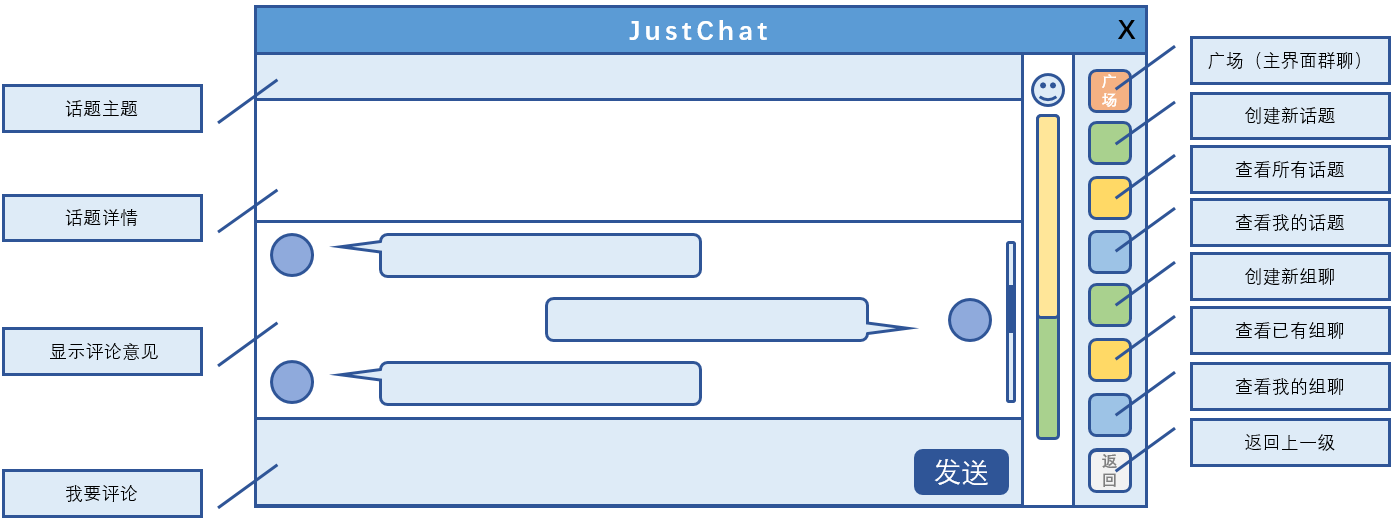
## 5.3查看所有话题

在此页面可以点击某个话题详情进入具体的话题详情页



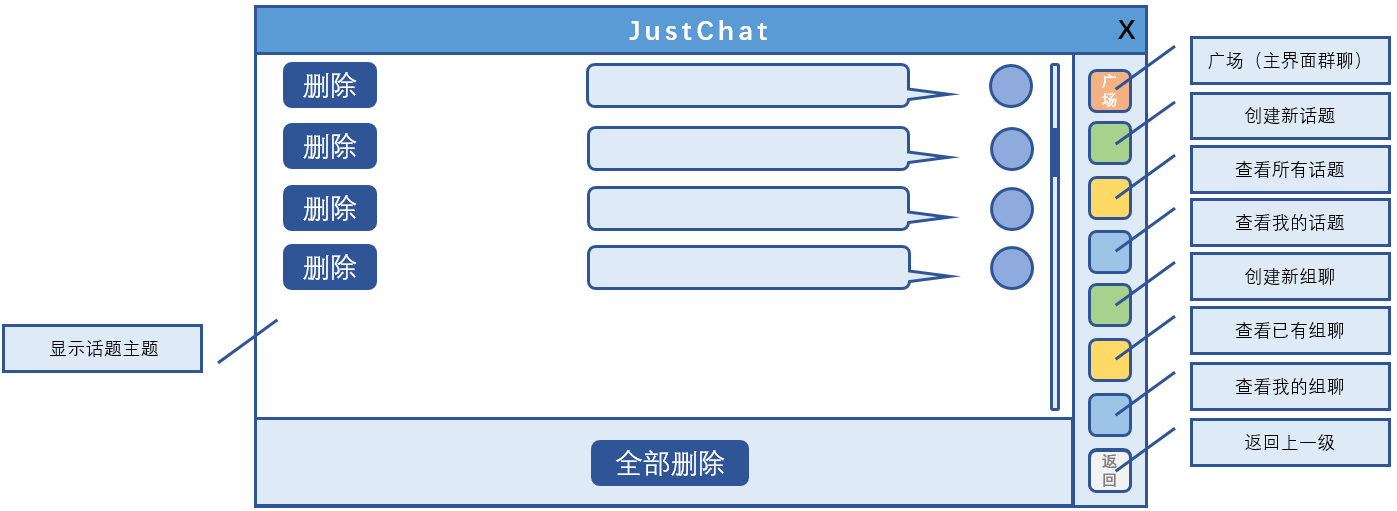
## 5.4查看某个具体话题

此页面右边的长条形即对当前所有评论态度的实时分析，即正面/中立/负面的情况，按百分比显示出来。



## 5.5查看我的话题

同样可以点击某个话题进入话题详情页。



## 5.6创建新组聊

类似创建新话题。

## 5.7查看所有组聊

类似查看所有话题。

## 5.8查看某个具体组聊

类似主界面的群聊界面，只是会在顶部显示组聊的名称。

## 5.9查看我的组聊

类似查看我的话题。

# 6情感分析功能设计

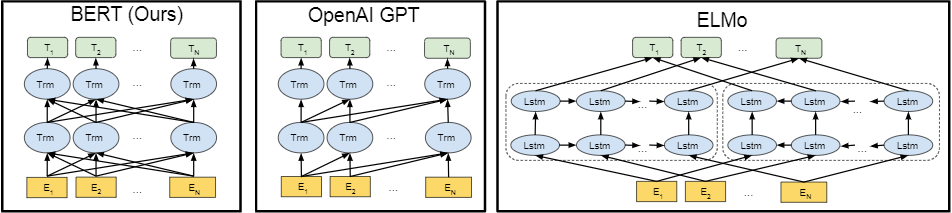
## 6.1介绍BERT

结合本P2P局域网聊天软件的特点，除了基本的通信聊天功能，我们将情感分析这一自然语言领域的分类任务用于软件之中，统计某一话题赞同和不赞同的人数，使得话题讨论更具意义。

2018年提出的BERT（Bidirectional Encoder Representations from Transformers）模型，使用多头注意力机制和位置嵌入，替换不易并行的循环神经网络，一举打破自然语言处理领域11个不同问题的最好记录。本文将采用BERT模型对本软件的话题聊天内容做中文文本分类。

BERT使用双向Transformer，在无标签文本上预训练深度双向表达，在每一层都受到左右文本的限制；OpenAI GPT使用从左到右的Transformer；ELMo使用独立训练地从左到右和从右到左的LSTM的连接来生成下游任务的特征。三者除了架构的不同，BERT和OpenAI GPT是基于微调的方法，然而ELMo是基于特征的方法。

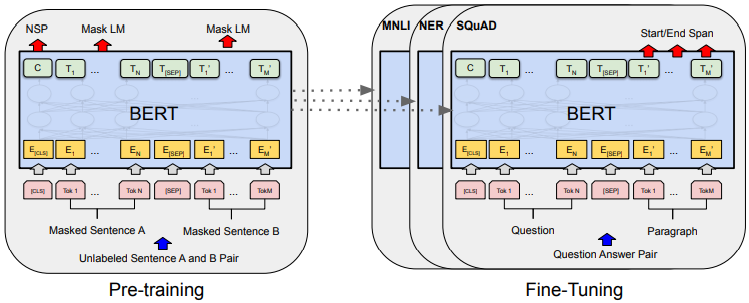
预训练模型的不同：



BERT训练分两个阶段：第一阶段采用双层双向Transformer模型，通过MLM（masked language model）和NSP（next sentence prediction）两种策略进行预训练；第二阶段采用Fine-Tuning的模式应用到下游任务。

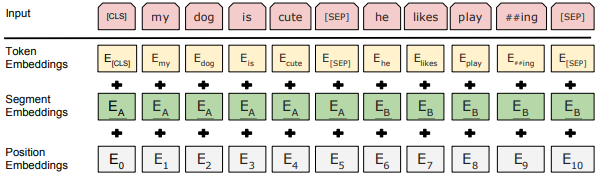
“完形填空”的学习模式迫使模型更多依赖上下文信息预测单词，赋予了模型一定的纠错能力；Transformer模型相比LSTM模型没有长度限制问题，具备更好的能力捕获上下文信息特征；相比单向训练模式，双向训练模型捕获上下文信息会更加全面。BERT一个独特的优点：对于下游不同的任务，都提供了一个统一的结构，即预训练网络结构和下游任务的网络结构差别很小。

BERT的预训练和微调过程：



预训练和微调网络结构仅在输出层不同，所以对于不同的下游任务即使用相同的预训练网络参数。微调时，所有参数都要微调。[CLS]是每个输入样本最前面的符号，[SEP]是特殊的分隔token（比如分隔问题和答案）。

BERT输入表达：



词嵌入由token嵌入、分隔嵌入、位置嵌入相加得到，从而BERT能够知道“句子”这个概念。Tocken嵌入是词向量，第一个单词是CLS标志，可以用于之后的分类任务；分隔嵌入用来区分两种句子，因为预训练包括了以两个句子为输入的分类任务；位置嵌入使得BERT能够生成查询向量和键向量。

## 6.2预训练BERT

BERT预训练中将传统的语言模型建模的任务划分为两个无监督的任务：完形填空（Masked LM）和句对预测（Next Sentence Prediction）NSP。

MLM:随机掩盖一定概率的输入tokens，然后预测这些masked tokens，即只预测这些掩盖的词语，而不是重构整个输入。

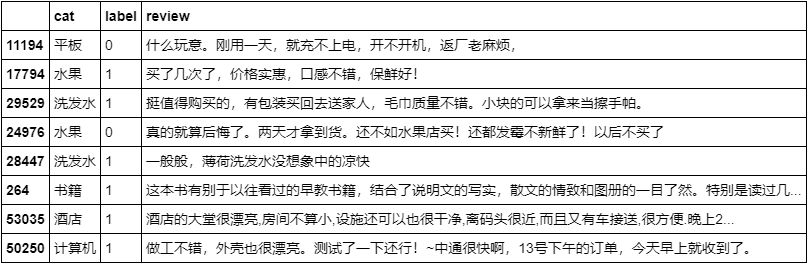
NSP: 许多重要的下游任务，如问答(QA)和自然语言推理(NLI)，都是建立在理解两个句子之间的关系的基础上的，而语言建模并不能直接捕捉这些关系。为了训练一个理解句子关系的模型，BERT预先训练了一个二值化的下一个句子预测任务。

## 6.3微调BERT

对于每个任务，只需将特定任务的输入和输出输入BERT并对所有参数进行端到端的finetune。

## 6.4基于BERT的情感分析

训练数据：从网上下载的网上购物情感极性分类数online\_shopping\_10\_cats（来自各电商平台），共有10个类别、6万多条评论数据，分别关于书籍、平板、手机、水果、洗发水、热水器、蒙牛、衣服、计算机、酒店，正、负向评论各约3万条。



如之前章节介绍，使用BERT的预训练模型，并在新数据集上进行微调可以得到很好的任务结果。本软件的情感分析即是中文文本的二分类问题，而在中文上BERT提供了预训练模型，所以我们能很快的利用预训练模型在本数据集上进行微调，得到本软件的话题情感分析模型，对某一话题下的评价进行正向和反向的情感统计。

# 7出错处理设计

不论硬件或软件，在运行过程中都会有出错的情况，这种错误可能来自于程序本身的BUG，也可能是用户操作的失误所造成的。当有错误发生时，我们应该有一个很好的机制来保障错误能够及时地被排除和解决，因此，当出现了错误的时候我们就需要程序能提供给我们出错的信息，这样用户就能够很快的找出具体的出错原因，以便寻找合理的途径去解决它。

本软件应根据不同的出错情况给出不同的出错信息，一般用对话框给出。对于一般错误，给用户提示信息，例如输入信息不符合要求，则会提示用户重新输入。对于查询错误，则会要求重新输入或退出操作；对于比较严重的错误，软件则会自动退出，向用户发出警报，并自动保存现场数据于日志中，恢复然后启动。

# 8安全保密设计

暂无。

# 9维护设计

暂无。