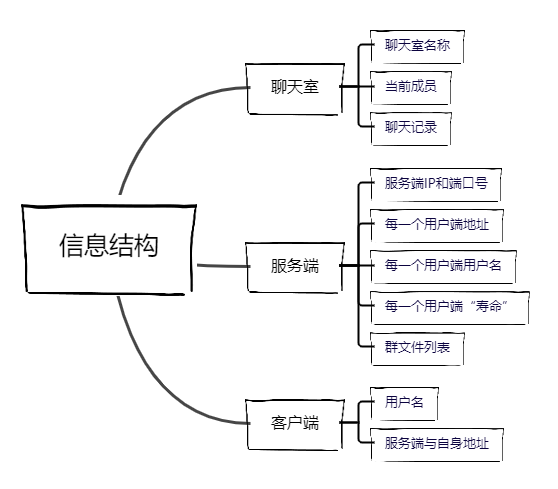
**Socket编程实践报告**

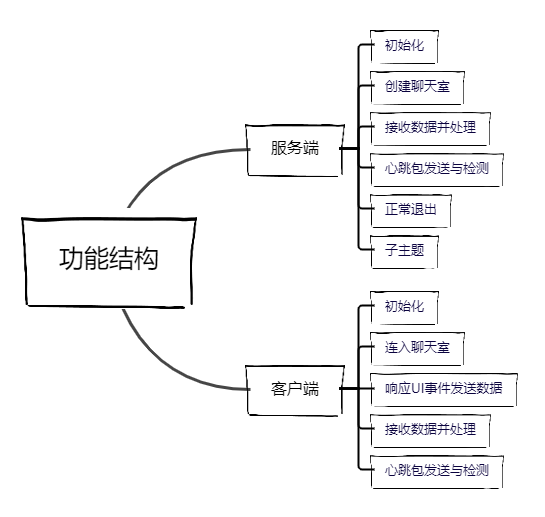
1. **需求分析**
2. 背景

随着互联网的发展以及广泛的普及，网络聊天工具已经日益成为人们日常交流的重要聊天工具。它成本低、交互性强、方便信息的交流和资料的传递。目前，应用广泛的聊天工具有QQ、微信、MSN等，这些软件深得大家的喜爱。在计算机网络课程实验中，我们将基于socket开发一个简易聊天室。

1. 目的
2. 理解TCP和UDP socket基本原理，**掌握socket编程技能**；
3. 理解网络应用层协议架构的基本原理，掌握相应架构设计和实现方法；
4. 提高**网络应用层协议设计**和实现的能力，提高撰写报告的能力。
5. 系统架构

系统以经典的**Client/Server结构**为设计标准。系统**基于socket通过自定义的消息结构**进行网络通信。服务端存储聊天室中用户和资源信息，并执行信息广播功能。用户可以上传和下载文件并多人聊天。





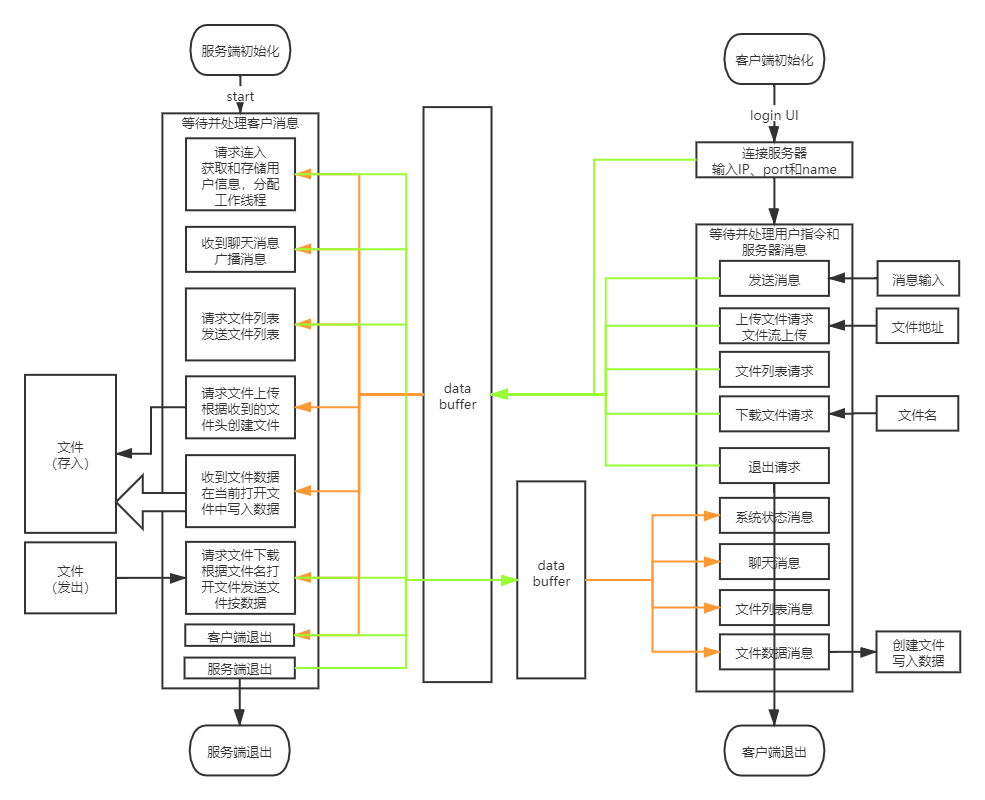
1. 功能需求

|  |  |
| --- | --- |
| 基本功能 | 是否完成 |
| 服务端可以建立网络聊天室并等待客户端连接 | 是 |
| 客户端可以连入网络聊天室，并成功将自己的消息发送至服务端 | 是 |
| 多个客户端能及时收到并显示彼此的聊天消息 | 是 |
| 新用户加入和旧用户退出不会影响聊天室运作 | 是 |
| 扩展功能 |  |
| 系统能正确处理粘包、掉线等异常情况 | 是 |
| 系统进行了高并发优化，并对其并发性能开展了评价实验 | 是 |
| 系统支持文件传输功能 | 是 |
| 系统支持语音聊天功能 | 否 |

1. 性能需求

在设计实验中，模拟2000个用户的并发连接和并行执行。预期指标：

1. 内存占用小于100MB
2. CPU利用率小于15%
3. 线程数量小于10
4. 服务端响应时间小于100ms（人能感知时间变化的最长时限为140ms）
5. 客户端无意外掉线，服务端不崩溃，系统平稳运行
6. **系统设计**

**服务端：**

* 服务端初始化时，根据预设的IP地址和端口进行监听，等待客户端进行连入
* 服务端创建四个工作线程（图中以其中一个工作线程为例），每个线程进行一个工作循环（等待并处理客户消息）。当有新用户连入时，将其按次序循环的分配给四个工作线程。
* 在工作循环中，线程通过select得知有用户消息，从用户的databuffer中取出并拆分，**根据消息内容执行不同的功能，功能模块执行成功后返回工作循环**。（各个具体功能实现后面会详细说明）
* 在实际编程中，我们还多创建了一个发送心跳包的线程（为结构图简洁性省略），用来进行客户的心跳包发送和“寿命”检测工作。
* 服务端在任何时间收到关闭聊天室的指令后，都回停下当前任务（在“主线程”中关闭所有工作线程），向所有的客户端发送服务端正常关闭通知，之后主线程也结束运行。

**客户端：**

* 客户端初始化时，先在登录界面输入自己的用户名，以及用户希望连入的聊天室地址，之后客户端根据这些设置发出连入的的申请。
* 客户端在工作线程中循环等待处理用户指令和服务端消息。**收到用户指令后，执行相应模块将指令内容打包发送到服务端；databuffer收到服务端消息后，对消息进行拆分、处理和展示，执行成功返回。**（功能具体实现见后）
* 除了一个工作线程以外，客户端也有一个心跳包发送线程。
* 收到用户退出指令时，客户端向服务端发送退出通知后退出。

1. **协议设计**
2. 数据包头部字段及格式

|  |  |
| --- | --- |
| 数据包头 | 数据主体 |

数据包头(8bytes): 消息类型(4bytes) + 消息长度(4bytes)

消息类型对应表格

|  |  |  |
| --- | --- | --- |
| 消类型息编号 | 服务端 | 客户端 |
| 0 | 客户端请求和状态消息 | 服务端状态和资源更新消息 |
| 1 | 客户端聊天消息文本 | 服务端广播的他人消息文本 |
| 2 | 客户端用户名 | 服务端返回的文件列表 |
| 3 | 心跳包问和答消息 | 心跳包问和答消息 |
| 5 | 文件头消息 | 文件头消息 |
| 6 | 文件数据流 | 文件数据流 |
| 7 | 客户请求下载的文件名 | —— |

数据包格式设计演变

|  |  |
| --- | --- |
| 数据包格式 | 分析 |
| 直接发送 | 优点：设计简单  缺点：易发生粘包和半包现象；无法区分消息类型，消息处理混乱；需预定义消息接收顺序 |
| 固定包长的数据包  规定：数据包长度为1024，接收方每次接收1024； 消息区分方式：预定义消息顺序，通过循环接收聊天消息，后续的特殊消息通过约定内容或是中断循环的方式接收，较为混乱 | 优点：解决了粘包和半包问题，实现简单  缺点：消息类型依旧难以区分，需要预定的消息传递顺序进行消息传输，接收方实现困难；对于短消息造成传输资源浪费 |
| **数据包头+数据体 -> databuffer**  发送方发送消息时，将消息体通过datapack加头打包为上述格式发送  接收方收到消息，先进行消息的拆包，后面根据消息类型选择运行。 | 优点：在应用层进行拆包和分类，解决了粘包和半包问题；根据消息头执行不同的功能，程序逻辑清晰；消息长度预知，增加了8字节包头却减少了不必要的消息体空位浪费  缺点：databuffer实际上是有极限的，若单个用户发送速率极大则可能会数据溢出；增加了数据包包头的传输和处理 |

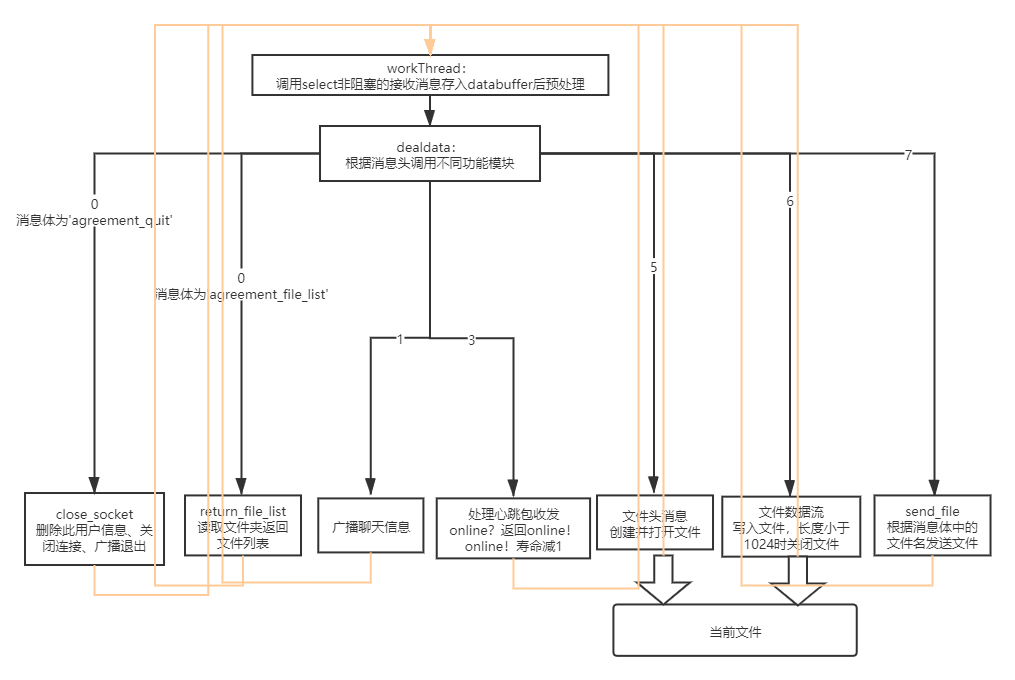
1. 协议规则（各模块的协作方法）
   1. 服务端与客户端的通信规则

统一的消息传输格式为上述的数据包头+数据主体格式。

发送方：将类型，消息文本传入packdata函数进行消息包装，然后发送。

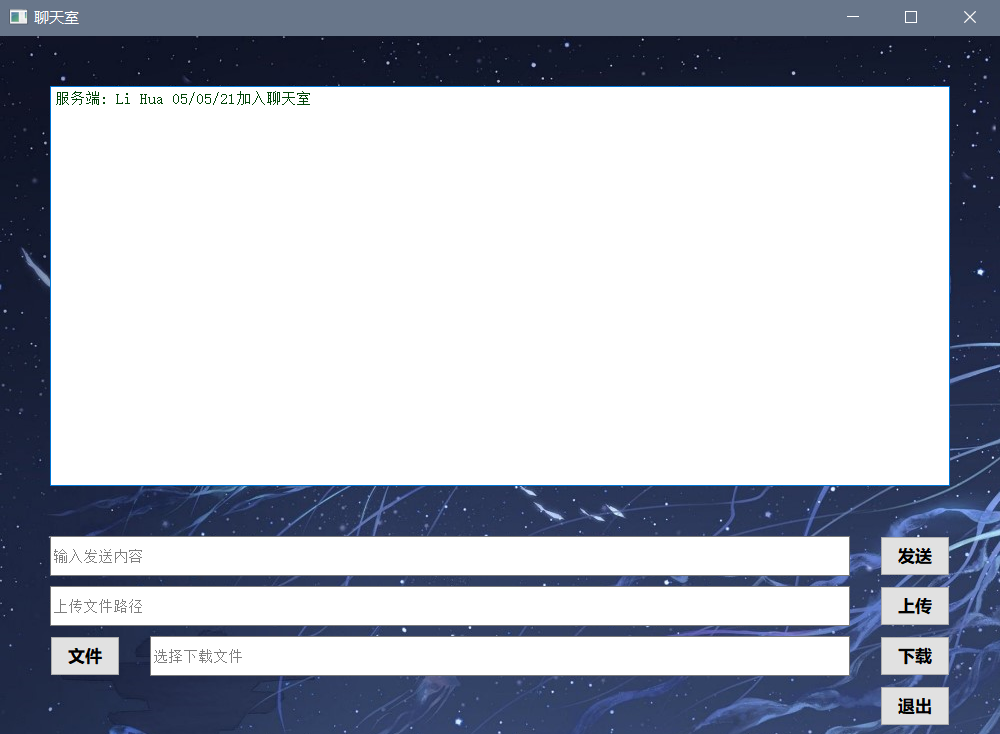
接收方：从databuffer中取出消息，提取消息头和消息体，根据消息头的中 类型调用不同的功能模块。

* 1. 服务端模块协作方法

1. 首先服务端主线程中进行初始化，通过UI由操作者控制开启（start）和关闭（destroy）。
2. 然后创建四个工作线程，每个线程独立处理一批用户，互不影响。单个线程中运行模块协作如图所示（框内英文字符串为实际函数名称，若没有则模块功能较为简单在case语句中处理）
3. 此外心跳包线程（heart）中进行心跳包发送与寿命检测。
4. 最后，用户点击的destroy时关闭服务端。
   1. 客户端模块协作方法

客户端的功能选择主要通过UI按钮触发

1. 首先客户端创建登录界面，用户输入IP、name、port
2. 然后客户端根据用户输入信息创建tcp连接并发送用户名，初始化用户变量。创建UI聊天界面，在聊天界面中进行控制。



创建线程监听按钮事件（功能可以同时多次使用，占用资源多，并发中利用精简客户端进行测试）

|  |  |
| --- | --- |
| 线程名 | 功能 |
| btn\_send | 监听发送按钮点击事件，从文本框获取消息发送 |
| btn\_upload | 监听上传按钮点击事件，从文本框获取地址上传文件upload\_file |
| btn\_download | 监听下载按钮点击事件，从文本框获取文件名发送文件下载请求（文件数据接收在recv\_msg中进行） |
| recv\_msg | 接收消息存入databuffer，拆分处理（过程简单，case中完成）  A>展示信息：分类颜色展示  B>文件信息：创建文件写入文件处理文件 |

1. 此外心跳包线程（heart）中进行心跳包发送与寿命检测。
2. 点击退出按钮，触发quit函数，通知服务端后退出程序。
   1. 协议规则设计原因

|  |  |
| --- | --- |
| 协议设计 | 分析 |
| 服务端：多路复用+多工作线程  用户将资源请求交给系统后等待，系统（资源透明）在资源可用时通知用户  建立四个工作进程，当用户连入时，循环的分配给四个工作线程的select列表中  在每个工作进程中，通过select返回所需资源空闲的socket请求列表。但是select只会将可处理socket放入预设数组，所以需要进行for循环查看是哪个socket可被处理。锁定sokcet后通过资源的状态采取接收数 | 四个工作线程，每个工作线程处理一批用户，可以防止windows系统因一个线程占用过多资源而杀死线程，使可处理用户数成倍增加；  系统处于非阻塞状态，不循环检测用户发来消息，消息来时放入才进行处理，减少了创建线程数从而增强了服务端的可并发性和稳定性。 |
| 客户端：功能线程 | 客户端无高并发需求，每个用户多个线程是可行的。不同的功能通过不同的线程执行，因此客户端可以同时执行不同功能（例如同时发消息和上传文件） |

1. **实现方法**

该网络聊天室是在Windows系统下， 使用Python语言的socket库实现基本功能，Tkinter库和QT库实现UI界面优化，服务端代码和客户端代码皆是单个文件运行，调用相关同文件夹下图片进行UI布局。

1. **性能评价**
2. 系统的可运行性评价

|  |  |
| --- | --- |
| 基本功能 | 是否完成 |
| 1.服务端可以建立网络聊天室并等待客户端连接 | 是 |
| 2.客户端可以连入网络聊天室，并成功将自己的消息发送至服务端 | 是 |
| 3.多个客户端能及时收到并显示彼此的聊天消息 | 是 |
| 4.新用户加入和旧用户退出不会影响聊天室运作 | 是 |
| 扩展功能 |  |
| 1.系统能正确处理粘包、掉线等异常情况 | 是 |
| 2.系统进行了高并发优化，并对其并发性能开展了评价实验 | 是 |
| 3.系统支持文件传输功能 | 是 |
| 4.系统支持语音聊天功能 | 否 |

该聊天室实现过程中，完成实现了基本功能和前三项拓展功能，在线下展示过程中流畅无bug出现。

**基本功能1**中我们为服务端设计了start和destroy功能，也就是创立聊天室进程和结束聊天室进程，且在结束时会为所有客户端发送提示信息，如遇到意外断网、程序崩溃等情况，客户端也会收到提示意外终止的消息提醒。

**基本功能2**中，客户端连入网络聊天室的过程中，服务端会为所有已在聊天室的用户发送新用户进入提醒。

**基本功能3**中，主机用户和其他用户之间的消息有颜色区分。

**基本功能4**中，新用户加入和旧用户的退出不会影响聊天室运作，且都会有提示。

**拓展功能1**里，实现了粘包­、半包和掉线等异常情况处理，解决办法：**a.粘包半包问题**：在服务端和用户端都设置databuffer缓冲区，消息由8字节列表的消息头[type,bodysize]和消息体（body消息内容）组成，以服务端运行为例，当消息到来时，先进入用户的dababuffer缓冲区。当databuffer中有消息时，先进行初步的消息提取，然后将type、bodysize、body传入dealdata函数进行消息处理。这样解决了粘包问题：无论粘包现象是否发生，消息都是先全部存入databuffer，然后根据消息头在应用层实现精确拆包；解决了接收方预设长度不够导致消息分割问题：虽然接收数据每次只接收1024，但是全部存入databuffer以后，被分割的消息又会重新连到一起。databuffer中消息依赖消息头分隔。**b.异常处理**：以服务端为例，通过“心跳包”来维护每个用户的“寿命”。服务端间隔一定时间向客户端发送心跳包，然后将客户端的“寿命”加一，若客户端及时回复约定消息，即将该客户端的“寿命”减一。当客户端的“寿命”达到10时，说明他大概率异常退出，则调用colse\_socket移除该用户。该拓展功能测试和展示中皆运行效果良好。

**拓展功能2**中，我们对其进行了高并发测试，可以实现2000用户同时进入聊天室发送接收一条消息后，每秒50个新用户进入且发送接收一条消息，延迟稳定在10ms以下。

在初始版本中，每有一个用户加入连接，则为该用户创建一个线程循环接收消息（甚至为每个用户创建一个发送心跳包的线程），在locust压力测试中出现了回复时延高、内容占用率高的问题。解决办法为多路复用方法+多工作线程，即用户将资源请求交给系统后等待，系统（资源透明）在资源可用时通知用户：**1> 建立四个工作进程**，当用户连入时，循环的分配给四个工作线程的select列表中；**2>在每个工作进程中，通过select返回所需资源空闲的socket请求列表**。但是select只会将可处理socket放入预设数组，所以需要进行for循环查看是哪个socket可被处理。锁定sokcet后通过资源的状态采取接收数据或是处理数据操作。该实现方法也有其局限性，每次select只能监听1024个文件描述符的事件，每次select返回都要进行对工作线程中所有socket进行线性扫描。

**拓展功能3**中，可查看所有用户上传的文件列表，更方便下载。实现过程：**1.文件上传**：文件头发送（这里的文件头作为消息的主体，消息类型为文件头类型，对方据此收到文件传输指令）。对方将文件头unpack后，通过文件头中的filename和filesize创建并打开文件。文件数据流传输时，每次发送定长的文件数据作为消息主体。消息头中类型为文件数据流类型。对方收到后，将数据流写入刚才打开的文件。**2.文件列表**：用户发送消息头为类型为文件列表类型的文件时，服务端返回文件列表。**3.文件下载**：接收方收到文件头消息后，根据文件头类型消息头判定要开始文件传输，进行文件创建和打开。然后接受文件数据流并写入该文件。当某次传输数据流小于约定长度时，说明文件传输完成，则关闭该文件并广播新文件到来。

其局限性在于文件上传时，持续的上传数据，无法进行后续数据的处理，因此可能造成未及时回复心跳包被错判死亡。因此这里双方约定在对面文件上传（自己文件下载）时，不发生心跳包。 这个问题在文件下载时是不存在的，因为文件下载时即使心跳包混入文件流包中，通过databuffer和消息头机制，也能正确拆包和处理。文件传输速度慢，大文件传输可能失败。

1. 性能评价

根据需求文档中自行选择和设置的性能指标，设计实验，评价所实现系统的各项性能。根据个人选择完成的任务数量不同，性能评价指标包含并发性能指标、稳定性能指标等内容。

需要注意所有评价实验应运行不少于3次后取平均值。性能评价部分不能直接罗列各项实验结果，而应通过表格、柱状图等形式对结果进行可视化，然后进行总结和分析。