河海大学物联网工程学院

课程设计报告

基于 Swing 和 Socket

在线聊天程序

学年学期	2022-2023 第二学期
学号姓名	1961010516 秦骁
课程名称	高级程序设计实践
授课班号	c0601545
授课教师	吕嘉

目录

一.分析和设计

1.1 基于模型-视图的整体结构设计

1.1.1 设计概述

该聊天程序基于模型-视图的整体结构设计，如图 1.1 所示包括三个部分，即模型（Model）、视图（View）和控制器（Controller）。模型是聊天应用程序中处理数据和业务逻辑的部分，是程序中所需数据的抽象表示，包括对数据进行读取、存储和更新的方法，以及处理业务逻辑的方法。模型通常会与数据库或其他数据源进行交互，以获取并处理数据，模型通过控制器来完成数据的传递。视图是应用程序中负责展示数据和与用户交互的部分。它是用户界面的抽象表示，负责将数据呈现给用户，并且支持用户与应用程序进行交互。视图通常包含了各种 UI 元素，包括聊天记录框、消息输入框等，视图仅负责展示数据和与用户交互，而不涉及任何业务逻辑。控制器是应用程序中协调模型和视图之间交互的部分。它接收来自视图的用户输入，并且根据用户输入更新模型，同时还会将更新后的数据返回给视图进行显示。控制器负责响应用户操作并调度模型和视图的交互。

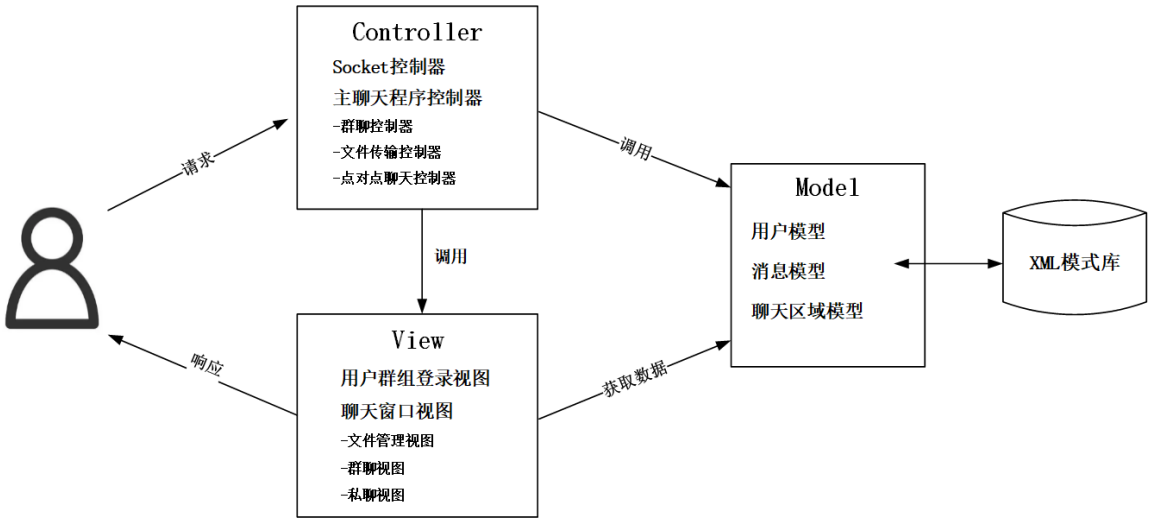

图 1.1 设计模式整体结构

1.1.2 模型设计

（1）用户信息类

保存用户的基本信息，该类的属性包括用户 ID、用户名、用户等级、用户性别。其中用户 ID 是唯一标识，将用户区分开来。

（2）消息类

保存每一条聊天记录的相关信息，该类的属性包括发送者编号、群组号、内容、时间。其中“发送者编号-群组号”表明该信息是由谁在哪个群组中发送的。内容是字符串，用于记录文字消息或者图片消息的 Base64 编码。时间为消息发送时系统的 ISO8601 格式时间。

（3）聊天区域类

保存聊天程序所属群组的聊天内容信息，该类的属性包括群组号和 Jrame 面板容器。其中 Jrame 面板容器为 JPanel 类，可以放入若干条 JTextArea 容纳的文字消息和 JLabel 容纳的图片消息。

1.1.3 视图设计

（1）用户群组登录视图

该视图为用户提供登录界面，用户输入用户 ID 和群组编号后，即可按该用户身份进入指定的群组。

（2）聊天窗口视图

该视图是程序的主界面的视图，子视图包括文件管理视图、群聊视图、私聊视图

文件管理视图的 UI 组件包括：可点击选中的供下载的文件列表、确认下载按钮、文件上传按钮。其中文件上传中会显示上传进度条，文件下载会有弹窗提示下载完成或失败。

群聊视图的 UI 组件包括：群聊历史记录显示区域、消息输入框、消息发送按钮、图片选择按钮。

私聊视图的 UI 组件包括：私聊历史记录显示区域、私聊对象选择下拉框、

消息输入框、消息发送按钮。

1.1.4 控制器设计

（1）Socket控制器

包括服务器和客户端，该控制器负责用户与服务器的连接管理，实时监测在线用户，现在点对点通信和广播。

（2）主控制器

该控制器作为整个聊天应用程序的入口点，并负责初始化应用程序的各个部分。它可以创建GUI视图对象，初始化网络连接和数据存储等其他对象，并将它们组合成一个完整的应用程序。

（3）聊天控制器

该控制器处理所有与聊天相关的逻辑，包括加载历史聊天记录、发送和接收消息、更新聊天记录等。它通过监听最新消息的记录时间来更新聊天视图并通知用户。

（4）文件传输控制器

该控制器负责管理所有与文件上传和下载相关的操作，并且向视图实时更新群组内可供下载的文件。它可以监视用户的文件上传请求，并将文件传输到服务器，当用户选中文件下载时，将文件下载到用户对应的数据文件夹里。

（5）点对点聊天控制器

该控制器处理用户选择私聊对象和发送消息的逻辑。加载私聊历史记录，向视图提供当时在线可被点击私聊的用户，监听用户选中想要私聊的对方，并提醒用户私聊视图的未读的新消息，完成用户发送和读取信息的逻辑。

1.2 功能模块划分

如图 1.2 所示，系统可以分为主要两个模块，分别是“多人聊天模块”、“点对点聊天模块”。“多人聊天模块”的功能细分为分组管理、文字聊天、图片聊天、消息历史、文件管理，“点对点聊天模块”的功能细分为在线监测、消息历史、未读提示、文字聊天。

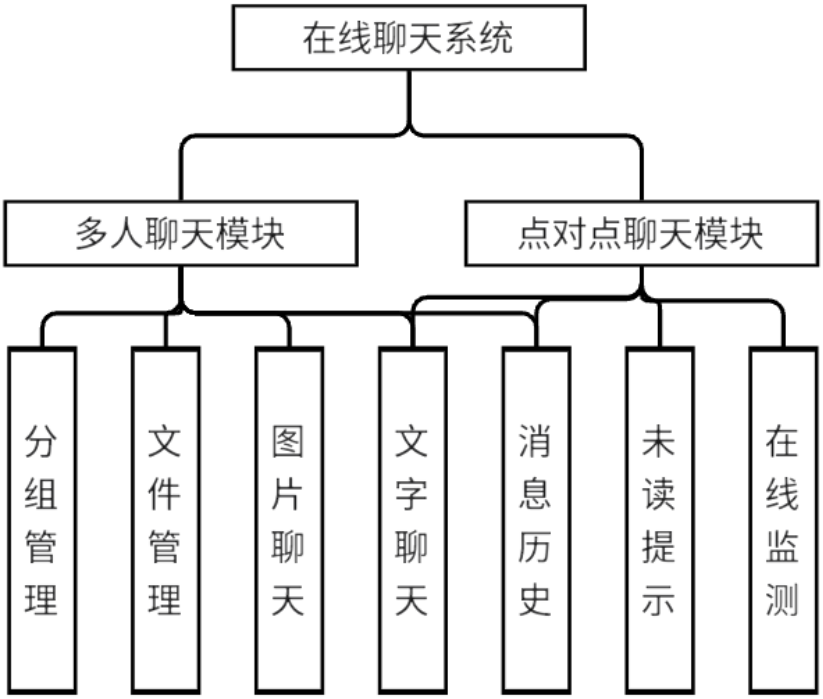

图 1.2 软件功能模块结构划分图

分组管理功能：用于控制用户在聊天软件中的聊天群组。用户在进入聊天界面前，先选择即将进入聊天的群组，在一个聊天群组中，用户只能看到当前群组的聊天信息，而不能接收到其他群组的聊天信息。

文件管理功能：用户可以在群组中上传或者下载文件。

图片聊天功能：用户可以在群组中发送图片消息。

文字聊天功能：用户可以在群组或者私聊中发送文字消息。

消息历史功能：用户在重新进入软件后，会重新加载聊天区域的历史信息。对于群聊模块，系统会加载该群组中历史聊天的所有文字和图片消息记录；对于点对点聊天模块，系统会加载当前使用系统的用户与该用户选中的对象的聊天历史的所有文字消息。

未读提示功能：私聊界面中，用户在接收到私聊对象的新消息时，私聊对象下拉框的对应用户名会变为红色，当用户点击查看消息之后，才会恢复默认状态。

在线监测功能：系统会监测当前群组的在线用户，用户只能选择当前在线的其他用户进行私聊。

1.3 类和对象分析与设计

系统的类和对象按如下类图 1.3 所示设计，在 1.3.1 和 1.3.2 小节详细阐述。

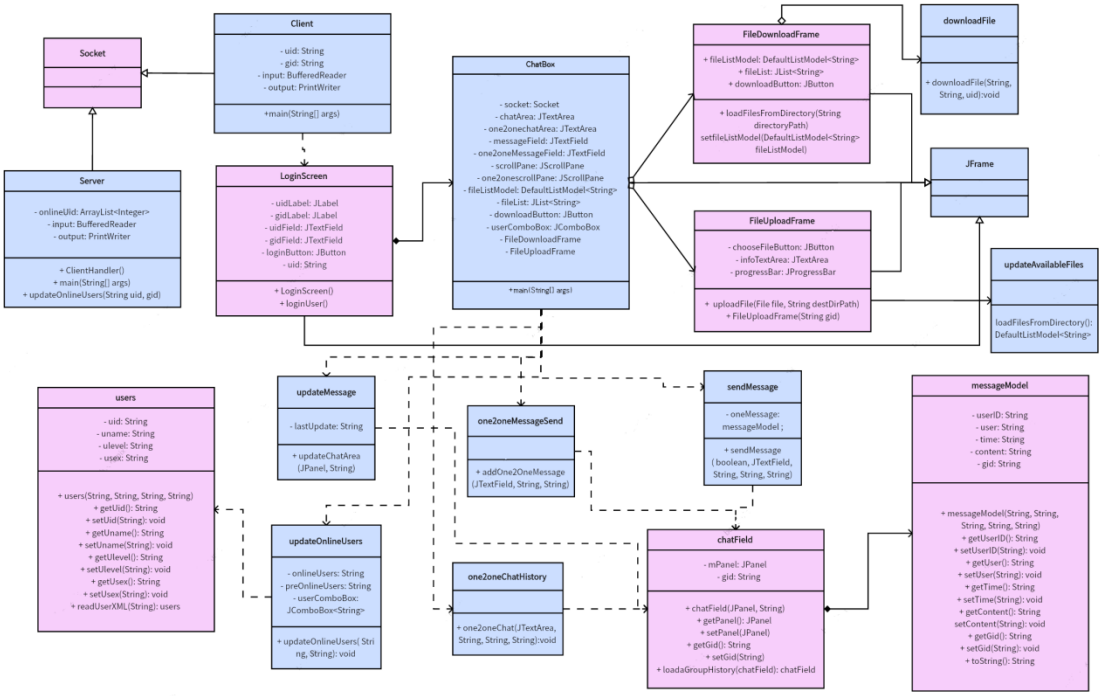

图 1. 3 系统类图设计

1.3.1 模型层设计

系统的模型层包括 user 类、messageModel 类、chatField 类。

user 类用于定义用户模型，成员变量包括用户 ID、用户昵称、用户性别、用户等级，方法包括各个属性的 get 和 set 方法，以及用于读取用户信息的解析用户 XML 数据库的方法。

messageModel 类用于定义消息模型，成员变量包括消息发送者 ID、消息发送者昵称、消息发送时间、消息内容、群组编号，方法包括各个属性的 get 和 set 方法，以及用于读取消息 XML 数据库并将消息的内容拼接的方法。

chatField 类用于定义聊天区域模型，成员变量包括 JPanel 容器和群组编号，方法包括各个属性的 get 和 set 方法，以及加载历史聊天记录的方法。chatField 由多条 messageModel 组成，因此两者为组合关系。

1.3.2 视图层设计

系统的视图层包括 loginScreen 类、chatBox 类、FileDownloadFrame 类、FileUploadFrame 类。其中 FileDownloadFrame 类、FileUploadFrame 类是 chatBox 的子视图。

（1）loginScreen 是系统的登录界面，组件包括两个输入框和一个提交按钮，输入框用于输入用户编号和群组编号，当点击按钮后，触发登录事件弹出 chatBox 界面，让用户进入指定群组。

（2）chatBox 是系统的主要程序界面，组件包括可滑动的容器 JScrollPane、群聊消息显示区域 JTextArea、私聊消息显示区域 JTextArea、群聊消息发送区域 JTextField、私聊消息方法送区域 JTextField、消息发送按钮 JButton、私聊对象选择栏 JComboBox、文件上传视图 FileUploadFrame、文件下载视图 FileDownloadFrame。群聊的消息区域在主界面的左侧，占据大部分位置，下方放置群聊消息输入框和发送按钮，当点击发送按钮后，输入框消息将会被发送至群聊。私聊的相关组件放置在主界面右上角，私聊对象选择框在最上方，下面依次是私聊消息显示区域、私聊消息输入框、私聊消息发送按钮，当点击发送按钮后，私聊消息输入框的内容将被发送至指定用户。文件上传下载的相关组件在剩余区域。

（3）FileDownloadFrame 和 FileUploadFrame 是 chatBox 的子视图，构成了系统的下载界面。视图包括可供点击的群组文件选择栏 JList 和下载按钮 JButton，当选择要下载的文件后，再点击下载按钮，系统会为用户下载指定文件。视图还有文件上传按钮 JButton 和上传进度条 JProcess，用户点击按钮后可以选择上传文件，确认上传后，系统会显示上传进度条告知用户上传进度。

1.3.3 系统过程分析与设计

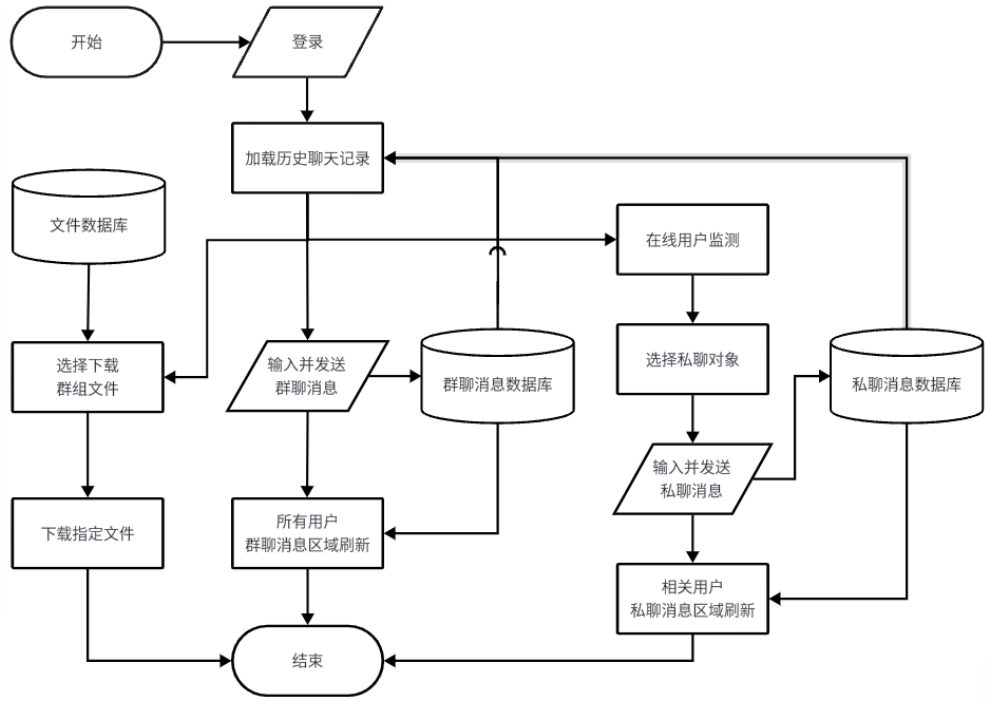

图 1.4 系统总体流程图

如图 1.4 所示，在用户登录进入程序主界面时，系统会自动加载聊天历史记录，包括群组聊天记录和私聊记录。之后，用户可以选择群组聊天、私聊或者下载文件。当用户在群组聊天框输入消息并且发送后，会将消息增加进群聊数据库，并且群组中当前所有在线用户都会接收到这条新消息。当用户想要发起私聊时，先选择在线用户下拉框中的私聊对象，然后在私聊输入框中写入消息并且发送后，会将消息增加进私聊数据库，并且对方会接收到这条消息。当用户选择文件并下载后，文件会传输进用户对应的文件夹中。

二.系统实现

2.1 系统应用的关键技术说明

2.1.1 Socket 通讯技术

在多人聊天软件中，Socket 通讯技术是实现客户端与服务器之间实时通信的核心。通过使用 Socket 类，您可以建立 TCP/IP 连接，并发送和接收消息数据。在 Java 多人聊天软件中，客户端通过 Socket 连接到服务器，并通过 Socket 对象进行数据的读取和发送。服务器端则监听客户端的连接请求，并为每个客户端创建一个独立的线程来处理与该客户端的通信。Socket 通讯技术使得用户可以实时地发送和接收聊天消息。

2.1.2 JFrame 框架

JFrame 是 Java Swing 库中的一个类，它提供了创建窗口应用程序的功能。在 Java 多人聊天软件中，通过使用 JFrame 框架，可以创建具有图形用户界面（GUI）的聊天窗口。可以使用 JFrame 类创建主窗口，并在其上添加文本框、按钮等组件，以便用户输入聊天内容和发送消息。JFrame 提供了丰富的布局管理器和组件，能够设计和构建用户友好的界面，提升用户体验。

2.1.3 Base64 编码

在多人聊天软件中，消息的传输需要进行数据编码和解码。Base64 编码是一种常见的二进制数据编码方式，它将二进制数据转换为可打印字符，方便在文本环境中传输和存储。可以使用 Base64 编码将文本消息或二进制文件进行编码，在网络传输或存储时使用。Java 提供了 Base64 类来处理 Base64 编码和解码的操作，使能够方便地在应用程序中使用这种编码方式。

2.1.4 XML 语言

系统利用 XML 技术来进行数据持久化。XML（可扩展标记语言，eXtensible Markup Language）是一种用于描述数据的标记语言。它通过使用自定义的标签来标记数据元素，并使用嵌套和层次结构来表示数据之间的关系。XML 允许以结构化的方式组织和存储数据。通过使用自定义的标签和属性，可以定义数据的结构和层次关系，使得数据具有更高的表达力和灵活性。XML 可以根据自己的需求定义自己的标签集合，并根据需要添加新的标签或属性。这使得 XML 适用于广泛的应用领域，从简单的配置文件到复杂的数据交换格式。XML 文档易于解析和处理，可以使用许多不同的解析技术进行处理和分析，可以使用 XML 解析器将 XML 文档转换为内部数据结构。

2.2 关键功能设计

2.2.1 群聊功能设计：

（1）消息存储

由于消息可以是文字的也可以是图片，所以要考虑消息类如何设计。本系统通过将图片转化为 Base64 编码的字符串存储，这样能够与文字消息一样统一存储，提高代码的复用性。

（2）加载消息历史

当用户重新进入群组时，系统通过 XML 解析技术读取文件中记录的消息历史。

（3）消息发送

通过监听消息发送按钮来触发消息发送事件，发送事件会将输入框视图的内容输入后续业务逻辑。通过监听表情发送按钮来触发表情图片选择事件，用户可以选择想要发送的表情，确定选择后输入后续业务逻辑。

（4）消息发送同步更新

通过实例一个消息类来向聊天区域视图新增消息，并且写入群聊历史消息记

录 XML 文件。为了实现所有用户都能接收到群组中任意用户发送的消息，利用一个 cache 来暂存最后一条更新消息时间，当检测到该消息时间与系统时间不同时，即认为有新消息，则为每一个进程里的用户的消息视图新增最新消息（这是在假设系统用户的并发程度不高的情况下采用的设计，当多用户同时刻发送消息，需要修改为“用户编号-时间”的检测或更为复杂的方式）。用户发送消息后，会向群组消息文件新增记录，用于对象反序列化恢复聊天历史记录。

（5）文件管理

系统只允许用户从当前群组中下载文件，因此要对用户在群组中可见的文件通过群组编号进行约束。系统为每个群组维护一个文件列表，当对应群组的文件新增或删除时，会动态地修改文件列表。用户可以向所登录的群组上传文件，系统提供一个上传进度条为用户提供友好的交互。用户可以从所登录的群组下载文件，文件将会下载到用户对应的数据文件夹。

2.2.2 点对点通信功能：

（1）在线用户检测

系统只允许用户对在线用户发起聊天，需要时刻监测群组中在线的用户，具体的实现方式如下：系统通过 socket 维护一个在线用户列表，当用户登录进入聊天群组时，系统会向对应群组的在线用户列表新增该用户编号；当用户退出聊天界面时，会触发事件让系统删除在线用户列表的该用户编号。

（2）私聊用户选择和通话

用户界面中，选择私聊用户的下拉框即是在线用户的名单，用户点击选择后，系统会修改该用户的私聊对象，并为用户刷新私聊界面。用户发送私聊消息后，会为自己和私聊对象的私聊文件新增记录，用于对象反序列化恢复聊天历史记录。

（3）未读消息提示

当用户发送私聊消息时，对方可能不知道有新消息发送，因此需要对接收私聊消息的用户进行提示。具体的实现方式是：为每一个用户维护一个私聊消息未读列表，当其他用户向该用户发送私聊消息时，将对应的私聊消息未读列表中的元素置为 1，并且私聊对象下拉选择框的对应项变为红色；当该用户点击该未读消息的发送方时，将对应的私聊消息未读列表中的元素置为 0，并且私聊对象下

拉选择框的对应项变回黑色。

2.3 关键用户界面设计

2.3.1 登录界面

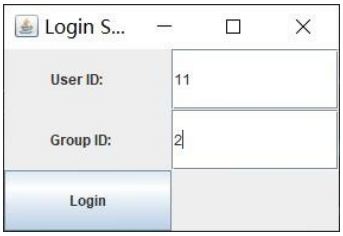

图 2.1 登录界面

该界面用于用户登录，可以输入用户编号和即将进入的群组编号，点击登录即可弹出程序的聊天界面。

2.3.2 聊天界面

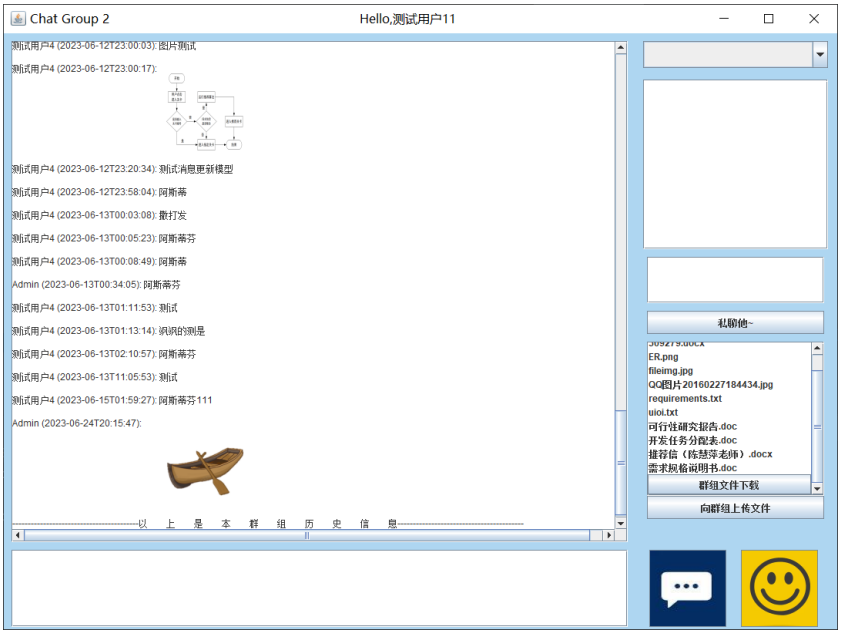

图 2.2 程序主界面

该界面是系统的主要界面，集成了多人聊天模块、点对点聊天模块和文件管理模块。左边是群聊区域，用户可以输入内容后点击深蓝色对话按钮发送消息；右上方是私聊区域，用户可以选择私聊对象并发送接收消息；右侧是文件上传和下载区域，可以选择文件下载或上传本地文件。

2.3.3 总体运行界面截图

本小节对系统的主要运行界面进行截图展示。

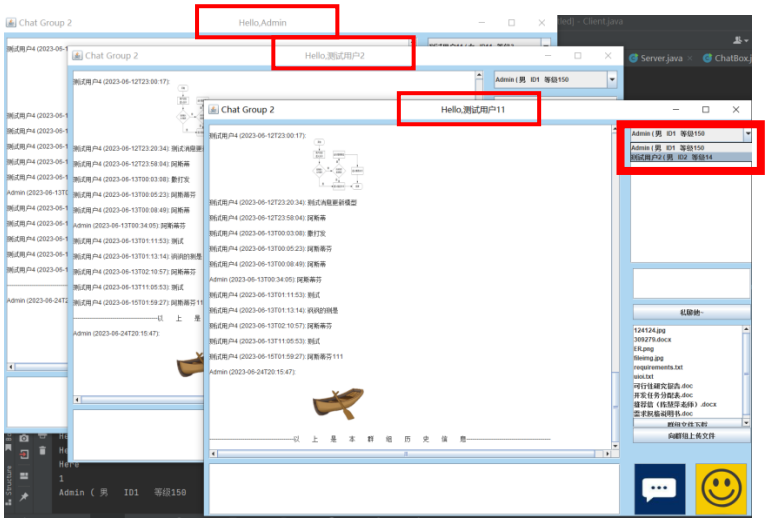

图 2.3 在线用户监测

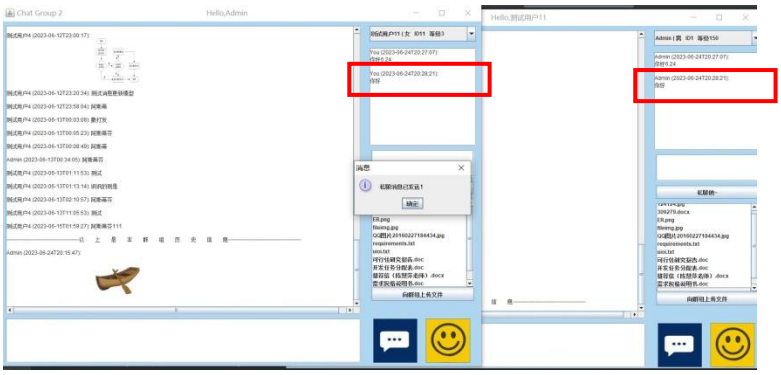

图 2.4 私聊消息同步

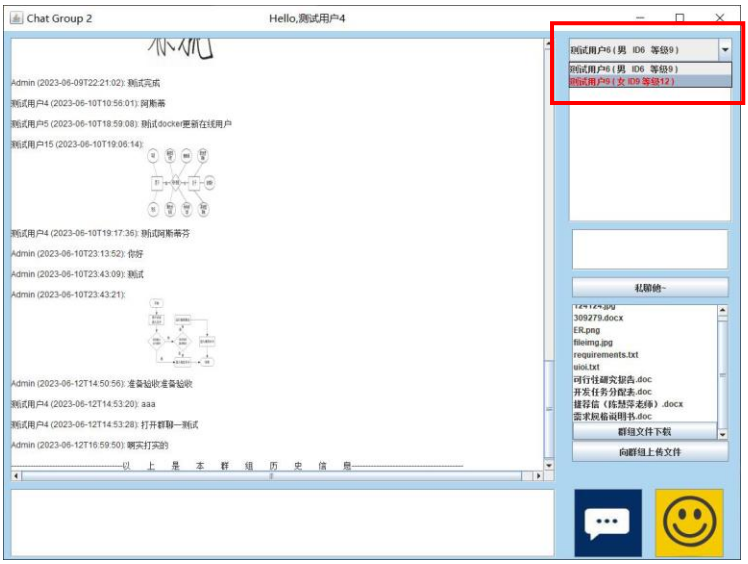

图 2.5 未读消息提示

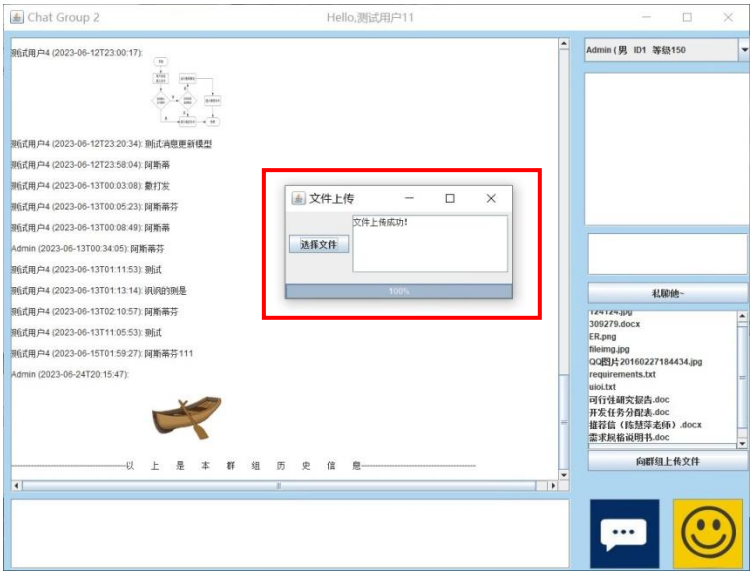

图 2.6 文件上传进度界面

2.4 关键业务类或者业务过程实现说明

（1）消息通信

```
Socket socket = new Socket("localhost", 9000);
// input是客户端从服务端接收的数据
BufferedReader input = new BufferedReader(new InputStreamReader(socket.getInputStream()));
// output是客户端向服务端发送的数据
PrintWriter output = new PrintWriter(socket.getOutputStream(), true);
BufferedReader consoleInput = new BufferedReader(new InputStreamReader(System.in));
OutputStream outputStream = socket.getOutputStream();
```

图 2.7 Socket 通信代码示例

如图 2.3 所示，服务器类使用 socket API 创建一个服务器套接字并监听指定端口。它能够接受来自客户端的连接请求，并维护一个客户端列表。当有新消息到达时，服务器应将其广播给所有连接的客户端。

```
// 将字节数组编码为 Base64 字符串
String base64String = Base64.getEncoder().encodeToString(bytes);
msg = "B64ENCODE" + base64String;
```

```
if (content.startsWith("B64ENCODE")) {
    // 消息内容content是图片编码，打印图片
    String result = content.substring(9);  // 删去前9个字符后转 Base64
      System.out.println(result);
    byte[] b64bytes = Base64.getDecoder().decode(result);
    // 读入字节数组为 BufferedImage 对象
    BufferedImage image = null;
    try {
        image = ImageIO.read(new ByteArrayInputStream(b64bytes));
    } catch (IOException ex) {
        ex.printStackTrace();
    }
```

图 2.8 图片消息 base64 编码方式保存和读取

图 2.4 展示了系统保存和加载图片消息的方式，利用 Base64 编码和解码 API。

（3）在线用户检测

```
if (log==-1){
    uid.remove(Integer.valueOf(-aa));
    System.out.println("Now uid list:" + uid);
    out.println(uid.toString());
}else{
    uid.add(aa);
    System.out.println("Now uid list:" + uid);
    out.println(uid.toString());
}
```

图 2.9 在线用户集合

当用户登录或退出系统时，都会更新系统的在线用户集合，系统通过在线 uid 集合来为用户提供私聊对象选择列表。

（3）私聊消息提示

```
public class BitArray {
    private int[] bits;
    private int size;

    public BitArray(int size) {
        this.size = size;
        int arraySize = (int) Math.ceil((size-1) / 32.0);
        bits = new int[arraySize];
    }
```

图 2.10 私聊未读数据结构

系统采用一个长度为在线用户数量减一的 0/1 数据结构来记录当前用户的未读私聊消息，0 的表示没有最新消息，系统采用默认黑色显示该私聊对象信息，1 的表示有未读的最新消息，系统标注为红色。当其他用户发送私聊消息时，会置相应位为 1；当用户点击该发送者时，会置相应位为 0。

三. 系统实施和部署

3.1 环境配置

系统的环境配置为 JDK1.8

3.2 系统运行

IDE 加载整个项目文件夹后，在\java\Controller\socketServer 下可以找到 Server.java，运行 Server.java 后，在同级目录下找到 Client.java，运行 Server.java 后，在 Server.java 的终端输入 run app，即可运行系统。每次输入 run app 都会打开一个新的程序界面。