**ChatRoom**

**详细设计说明书**

**编 写 人**：李益军

**编写日期**： 2019年 12月 2日

文档信息：

|  |  |
| --- | --- |
|  |  |
| 文档名称 | ChatRoom概要设计说明书 |
| 描述 | 根据架构设计确定的各子系统、模块，对其中的类和接口进行设计。 |
| 负责人 | 李益军 |
| 状态 | 编辑 |

**文档变更记录：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 版本编号 | 说明：如形成文件、变更内容和变更范围 | 日期 | 变更人 | 批准日期 | 批准人 |
| V0.1 | 形成文件并编辑 | 2019/12/2 | 李益军 | 2019/12/2 | 李益军 |
| V0.2 | 增加程序结构 | 2019/12/3 | 李益军 | 2019/12/3 | 李益军 |
| V0.3 | 增加通讯模块和数据库模块 | 2019/12/5 | 李益军 | 2019/12/5 | 李益军 |
| V0.4 | 增加服务器模块 | 2019/12/8 | 李益军 | 2019/12/8 | 李益军 |
|  |  |  |  |  |  |

目录

[1.引言 4](#_Toc26733549)

[1.1编写目的 4](#_Toc26733550)

[1.2背景 4](#_Toc26733551)

[1.3技术 4](#_Toc26733552)

[1.4参考资料 4](#_Toc26733553)

[2.程序系统的结构 5](#_Toc26733554)

[2.1程序系统结构 5](#_Toc26733555)

[2.2程序结构模块描述 5](#_Toc26733556)

[3．功能设计 6](#_Toc26733557)

[3.1通讯模块 6](#_Toc26733558)

[3.2数据库模块 9](#_Toc26733559)

[3.3 服务器模块 11](#_Toc26733560)

[3.4 客户端模块 15](#_Toc26733561)

## 1.引言

### 1.1编写目的

本文档为ChatRoom项目的详细设计文档，是对该项目进行详细设计，在概要设计的基础上进一步明确系统结构，详细的介绍系统的各个模块，为进行后面的实现和测试做准备。本阶段完成系统的详细设计并明确系统的数据结构和龚总流程。是将用户需求转化为软件表示的过程。

本详细设计说明书的预期读者为本项目小组的成员以及日后对该系统感兴趣的人员。

### 1.2背景

此系统名为：ChatRoom。是软件工程的课程项目。

### 1.3技术

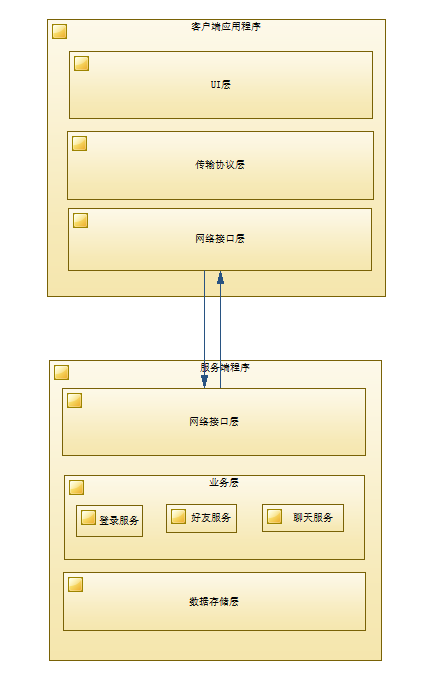
1. Client/Server: 软件结构
2. MySQL：数据库，存储数据
3. JAVA编程语言
4. PowerDesigner绘图工具

### 1.4参考资料

1. 需求分析报告
2. 概要设计说明书

## 2.程序系统的结构

### 2.1程序系统结构



### 2.2程序结构模块描述

2.2.1 服务端

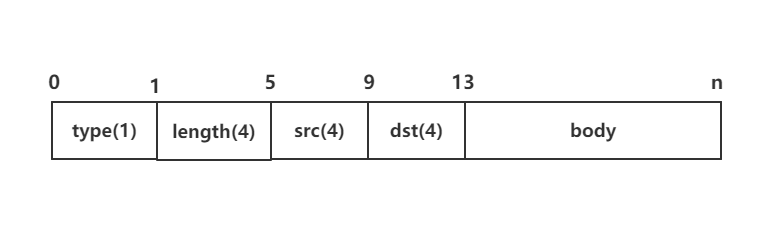
2.2.2 客户端

## 3．功能设计

### 3.1通讯模块

#### 3.1.1 消息格式

3.1.1.1 消息头（msghead）：



|  |  |  |
| --- | --- | --- |
| 属性名 | 类型 | 长度 |
| Type | Byte | 1 |
| length | Int | 4 |
| Src | Int | 4 |
| Dst | Int | 4 |

3.1.1.1.2 消息类型

|  |  |
| --- | --- |
| 类型 | 值 |
| 注册请求信息 | 0x01 |
| 注册应答消息 | 0x11 |
| 登陆请求信息 | 0x02 |
| .登录应答消息 | 0x22 |
| 添加好友信息 | 0x05 |
| 添加好友应答信息 | 0x55 |
| 好友列表 | 0x03 |
| 聊天信息 | 0x04 |

3.1.1.2 消息体（msgbody）：

3.1.1.2.1 注册消息

|  |  |  |
| --- | --- | --- |
| 属性名 | 类型 | 长度 |
| Userid | String | 10 |
| Username | String | 10 |
| Pwd | String | 10 |

3.1.1.2.2注册应答消息

|  |  |  |
| --- | --- | --- |
| 属性名 | 类型 | 长度 |
| State | byte | 1 |

3.1.1.2.3 登录消息

|  |  |  |
| --- | --- | --- |
| 属性名 | 类型 | 长度 |
| Userid | string | 10 |
| Pwd | String | 10 |

3.1.1.2.4登录应答消息

|  |  |  |
| --- | --- | --- |
| 属性名 | 类型 | 长度 |
| State | byte | 1 |

3.1.1.2.5 拉取好友列表消息

|  |  |  |
| --- | --- | --- |
| 属性名 | 类型 | 长度 |
| NULL | NULL | 0 |

3.1.1.2.6 好友列表应答消息

|  |  |  |  |
| --- | --- | --- | --- |
| 属性名 | 类型 | 长度 | 注释 |
| Listcount | byte | 1 | 好友分组个数 |
| Listname | String[] | 10 | 分组名称 |
| Friendcount | Byte[] | 1 | 本组有多少个用户 |
| Username | String | 10 | 好友名字 |
| Userid | int | 4 | 好友id |
| byte bodyState | byte | 1 | 好友在线状态 |

3.1.1.2.7 聊天消息

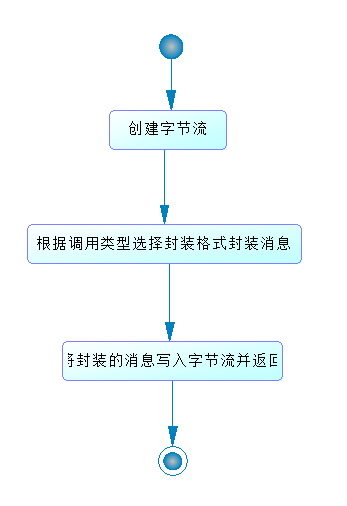
|  |  |  |
| --- | --- | --- |
| 属性名 | 类型 | 长度 |
| Text | String | 65536 |

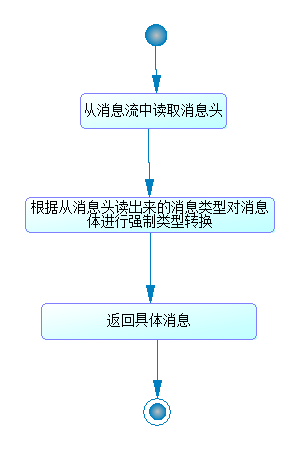
#### 3.1.2 过程描述与流程图

* 过程描述

1. 封装消息：客户端/服务器根据操作类型/收到消息类型，按照3.1.1中的消息格式，填充相应字段，最后得到一个包含特定消息的字节数组。
2. 解析消息：客户端/服务器将收到的字节流，按照3.1.1中的格式，先提取消息类型，再按照相应字段解封装，最后得到真实消息。

* 流程图





#### 3.1.3 限制条件和测试要点

* 限制条件

1. 要封装的数据数据类型必须在消息格式规定的类型之中。
2. 读取消息时必须保证读取的字节流长度和发送时计算出的字节流长度相等。
3. 必须严格按照消息格式封装。

* 测试要点

1. 重点测试消息先封装，在解封装后是否相等。

#### 3.1.4 接口

1. public static MsgHead parseMsg(byte[] data)
   1. 功能描述：解析收到的数据流
   2. 参数：byte[] data—收到的字节流
   3. 返回值： 解析出来的消息头
2. public static byte[] packMsg(MsgHead msg)
   1. 功能描述：根据消息类型打包消息
   2. 参数：MsgHead msg—包装好的消息头
   3. 返回值： byte[]—打包好的字节流

### 3.2数据库模块

#### 3.2.1 功能

* 功能

1. 连接数据库
2. 释放所占据的数据库资源
3. 通用的查询接口
4. 通用的更新接口
5. 通用的插入接口

#### 3.2.2 数据库实现

* userinfo

|  |  |  |
| --- | --- | --- |
| 属性名 | 类型 | 说明 |
| uid | int | 用户id,主键，自增 |
| pwd | string | 用户密码 |
| nickname | string | 用户昵称 |
| vatar | int | 用户头像 |

* lid\_uid

|  |  |  |
| --- | --- | --- |
| 属性名 | 类型 | 说明 |
| lid | int | 好友分组索引,主键，自增 |
| uid | int | 所属用户id |
| lname | string | 分组名称 |

* relationship\_1

|  |  |  |
| --- | --- | --- |
| 属性名 | 类型 | 说明 |
| uid | int | 用户id |
| lid | int | 所属分组索引 |

#### 3.2.3 限制条件和测试要点

* 限制条件

1. Relationship\_1中uid,lid分别引用userinfo中的uid与lid\_uid中的lid
2. userinfo中的uid与lid\_uid中的lid为自增类型，插入时无须显示指定

* 测试要点

1. 测试能否正常进行数据库的增删查改

#### 3.2.4 接口

1. public Connection getConnection()
   1. 功能描述：连接数据库
   2. 参数：无
   3. 返回值： 数据库连接对象（Connection）
2. public void release()
   1. 功能描述：释放数据库连接资源
   2. 参数：无
   3. 返回值：无
3. public ResultSet query (String sql)
   1. 功能描述：执行查询sql语句
   2. 参数：String sql—查询语句
   3. 返回值：返回查询结果--ResultSet对象
4. public int update(String sql)
   1. 功能描述：执行更新sql语句
   2. 参数：String sql—更新语句
   3. 返回值：返回更新影响到的条目
5. public int insertAndGet(String sql)
   1. 功能描述：执行插入一个新条目的sql语句并返回插入位置的索引
   2. 参数：String sql—插入语句
   3. 返回值：返回插入之后的索引

### 3.3 服务器模块

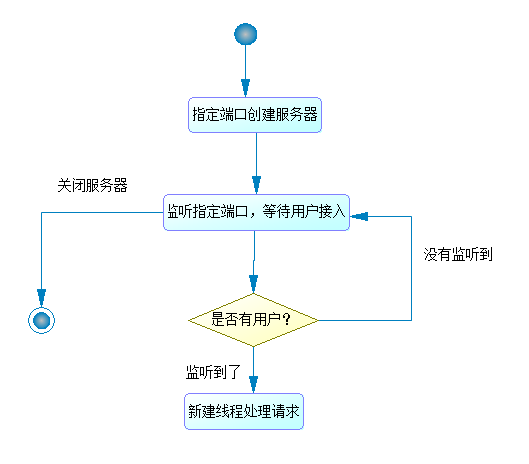
#### 3.3.1 功能

* 功能

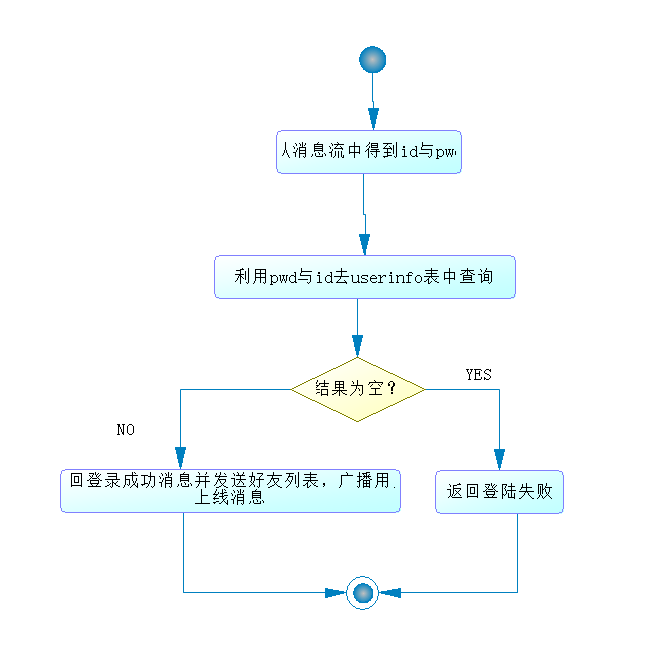
1. 创建服务器：创建一个服务器处理业务，新建线程处理客户请求
2. 响应客户端登录账号的请求：对每个登录请求做出响应
3. 响应客户端注册账号的请求：对每个注册请求做出响应
4. 响应客户端加载好友列表的请求：对每个客户端显示好友列表的请求进行响应
5. 支持一对一聊天：支持两个客户端之间进行一对一聊天

#### 3.3.2流程图

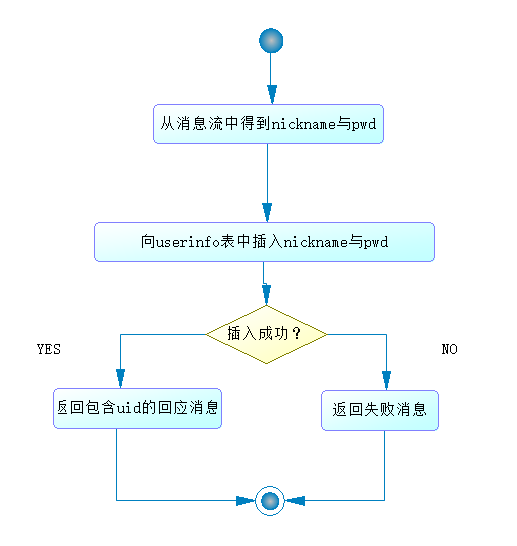
* 创建服务器



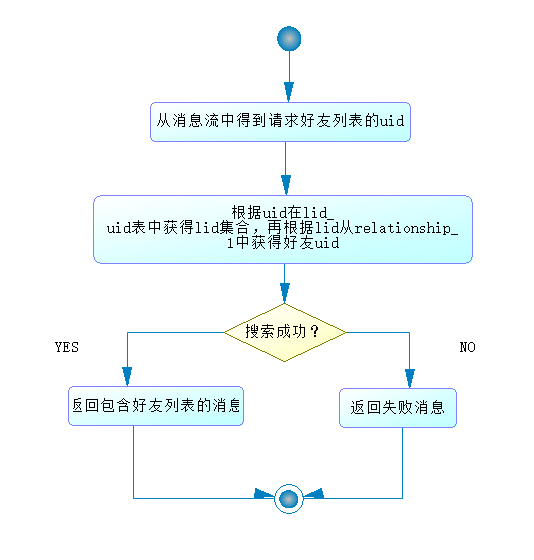
* 响应客户端登录账号的请求



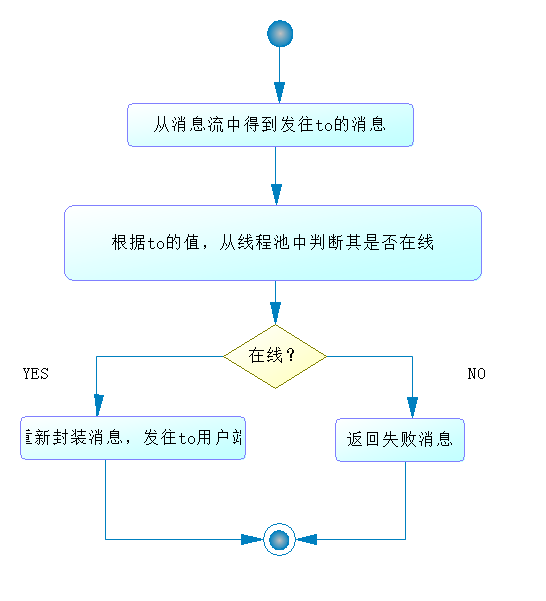
* 响应客户端注册账号的请求



* 响应客户端加载好友列表的请求



* 支持一对一聊天



#### 3.3.3 限制条件和测试要点

* 限制条件

1. 通讯的双方必须同时在线

* 测试要点

1. 按顺序测试每个动作，查看结果是否与预期结果符合

#### 3.3.4 接口

1. public void setupServer(int port)
   1. 功能描述：服务器启动
   2. 参数：int port—服务器监听端口
   3. 返回值： 无
2. public void closeServer()
   1. 功能描述：服务器关闭
   2. 参数：无
   3. 返回值： 无
3. private void processLogin()
   1. 功能描述：处理登录/注册信息
   2. 参数：无
   3. 返回值： 无
4. private void broadcastState()
   1. 功能描述：某个用户上线后向所有其他用户广播其上线状态
   2. 参数：无
   3. 返回值： 无
5. private void sendFriendList()
   1. 功能描述：向用户发送其好友列表
   2. 参数：无
   3. 返回值： 无
6. public void processChat()
   1. 功能描述：处理用户一对一聊天/添加好友消息
   2. 参数：无
   3. 返回值： 无
7. public void sendMsg(int from, String msg)
   1. 功能描述：向用户发送来自于from的消息
   2. 参数：int from—消息来源的uid

String msg—消息内容

* 1. 返回值： 无

### 3.4 客户端模块

#### 3.4.1 功能

* 功能

1. 链接服务器：确认客户端与服务器端可以通讯
2. 登录/注册账号：客户端能登录/注册聊天用的账号
3. 加载好友列表：能收到服务端发来的好友列表并显示；
4. 聊天：两个客户端能通过服务器进行一对一聊天

#### 3.4.2 流程图

* 登录/注册账号
* 加载好友列表
* 聊天

#### 3.4.3 限制条件和测试要点

#### 3.4.4 接口