# Client-Server 程序文档


同濟大學

项目组编号：03

成　　员：倪雨婷 1352837　罗晓丹 1352866

王刚 1352873　叶坤宇 1352895　刘旭东 1352918

时　　间：2016-3-28

## 文档信息：

| | | | |
|---|---|---|---|
| 文档名称 | 项目程序文档 | | |
| 版本 | 2.0 | | |
| 文档编号 | 01 | | |
| 文档状态 | [ ]草稿 [ ]正在修改 [ √ ]正式发布 | | |
| 项目名称 | Client-Server 应用程序 | | |
| 撰写 | 王刚 | 日期 | 2016/3/28 |

## 修订历史记录：

| 日期 | 版本 | 说明 | 作者 |
| --- | --- | --- | --- |
| 2016/3/26 | V1.0 | 撰写初稿 | 王刚 |
| 2016/3/28 | V2.0 | 进行更新修改，撰写终稿 | 王刚 |
| | | | |

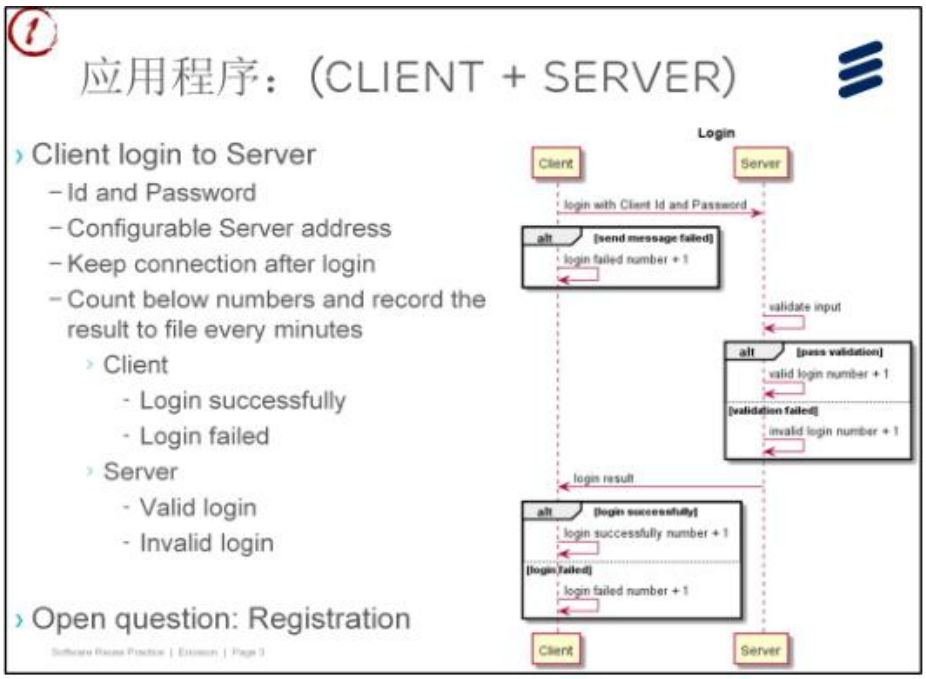
# 一. 引言

## 1.1 程序编写目标：

建立一个信息通信的应用，主要包括 Client 以及 Server 两部分的功能，具体来说将实现以下功能：

- ❖ 实现用户的登录和注册；
- ❖ 保持用户端和服务端的不断的 connection；
- ❖ 记录并显示客户端登录状态以及发送的信息；（Server 端会忽略未登录状态下 Client 发送的消息）
- ❖ 服务端将任意客户端的信息转发到其余登录状态下的客户端；
- ❖ 拒绝客户端在一秒钟内发送超过 5 则信息到服务端；
- ❖ 当某一客户端发送 100 则总数的信息后自动退出登录，下次登录发送信息数将从 0 重新计数；
- ❖ Server 端能够记录/忽略的消息数以及自身转发的消息数，Client 端能够记录该端发送的消息数以及接收的消息数；

## 1.2 项目具体成果要求：




# 二．程序设计

本项目为基于 Windows 平台，采用 Java 语言编写，使用中间件 ActiveMQ 进行通信的搭建。主要设计有界面有“登录注册界面”和“客户端界面”。总体功能上分为客户端部分和服务端部分。主要功能是客户端向服务端发送信息，服务端接收信息并转发到其余处于成功登录状态的客户端上显示。

## （一） Client 部分：

**提供登录注册功能：**



首先在 Server 中的类 User 中有关于用户各自信息的定义：

这里有关的是：

```
public  String UserName;
public  String Password;
```

通过遍历用户姓名，进行与输入框中的用户名的匹配，根据匹配结果再进行密码校对，从而确定登录的具体情况并在之后显示出来。

用到的组件有：

```
JLabel regUsername;
JLabel regPassword;
JTextField regUsernameInput;
JPasswordField regPasswordInput;
JButton regBtn;
```

此处“登录界面”的布局由下列代码布置：

```
username = new JLabel("用户名 :");
password = new JLabel("密   码 :");
usernameInput = new JTextField(10);
passwordInput = new JPasswordField(10);
login = new JButton("登陆");
register = new JButton("注册");
```

具体说来，当用户点击 Button “登录”的时候，会触发 `loginPanel.add(login);` 实现登录功能。

而点击“注册”按钮时候，会跳转到“注册界面”，此页面的布置与前面的登录界面类似。

登录执行之后会根据反馈结果进行相关的状态反馈：

- boolean checkConnection(String userName)

| 返回值 | 代表结果 |
| --- | --- |
| true | 在登录 |
| false | 未登录 |

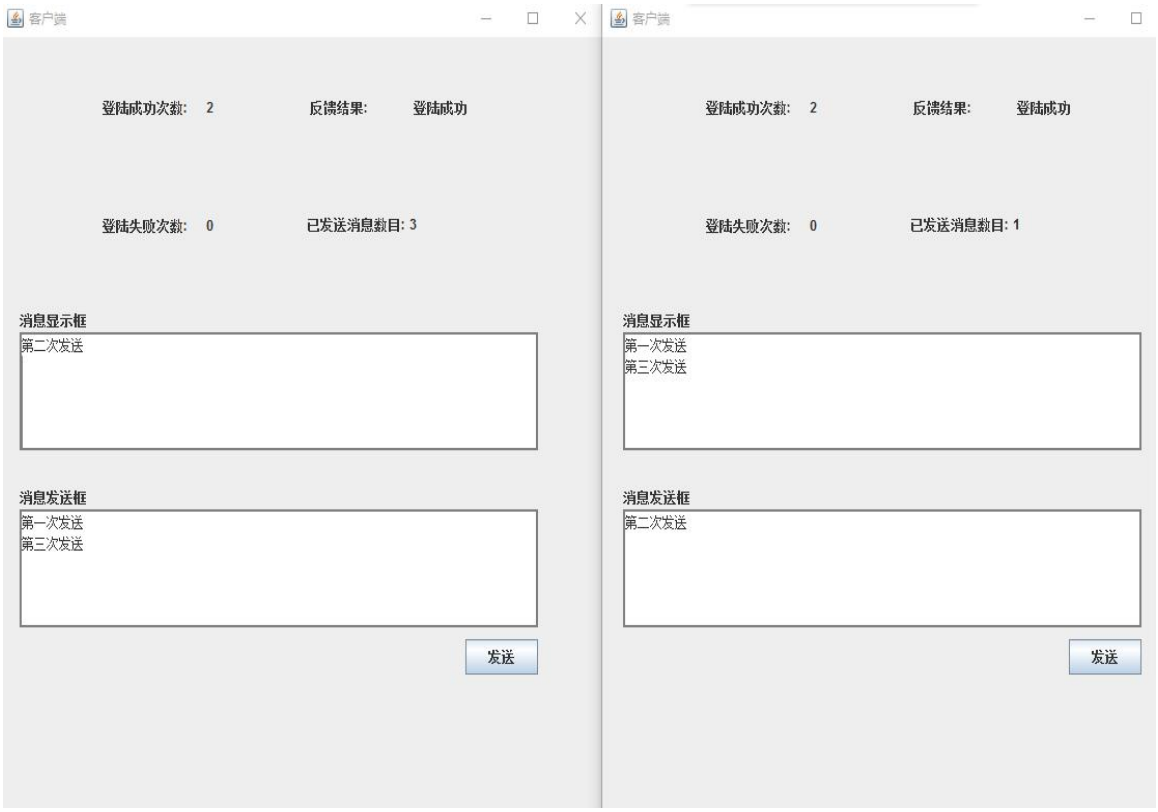

## “客户端界面”功能：

实现界面如下：

直观上可以看出，这里主要包含了“登录状态”显示、“登录成功次数”显示、“登录失败次数”的记录以及“已发送消息数目”的记录。

下方的两部分对话框类似的部件是用以执行“客户端向服务端发送消息”以及“服务端转发的消息显示”的两个窗口。其中，向服务端执行发送的功能由Button“发送”控制。

部分功能演示结果如下：




位于“客户端”界面上面部分的四个信息，是由 Client 处代码对 Server 端所记录的数据进行调用，并进行分析解释后显示出来的结果，下方为举例说明：

Server 端：

```
.......(前面代码省略)
    validLoginTime++;
    theUser.isLogin = true;
    return 200;
```

Client 端：

```
public void ListenMsg() throws JMSException {......}
```

.....(前面代码省略)

```
if(txtMsg.getText().equals("200")){
feedbackDisplay.setText("登陆成功");}
```

【注：Server 端返回结果 200 代表的是“登录成功”】

而消息发送功能【转发功能(在之后的 Server 端分析中将说明)】由以下代码实现：

消息发送功能：.....(前面代码省略)

```
sentButton.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
// TODO Auto-generated method stub
String str = msgSent.getText();
msgNumberCount++;
msgNumberDisplay.setText(String.valueOf(msgNumberCount));
//msgDisplay.setText(str);   //显示到消息显示框
status=false;
sendMsg(str,"Ericsson",false);
}
});
.............
// 定义 sendMsg()功能函数：
public  void sendMsg(String msgText,String toipcName,boolean isLogin){
.............
}
```

这样便实现了将“消息发送框”中的信息文本读取并发送给到服务端的功能。

## （二） Server 部分：

限制用户不得在一秒时间内发送 5 则以上消息：

```
if (current-user.loginDate < 1000)
{
```

```
if(user.loginRequsetTime<5)
{
    System.out.println(user.loginRequsetTime);
    user.loginRequsetTime++;
    return  true;
    }}
```

设定在用户发送消息总数小于 100 以及超过 100 时的应对：

```
sender.sendMessagesNum++;
System.out.println("Server 0");
if (sender.sendMessagesNum < 100) {
   System.out.println("Server 1");
   return true;
}else return false;
//超过 100 后退出重登录，并将发送消息数归 0
.............(前面代码省略)
sender.isLogin = false;
sender.sendMessagesNum = 0;
```

## 监听客户端发送的消息并转发：

### 监听客户端：

```
public void ListenMsg(){
     ActiveMQConnectionFactory factory = new
ActiveMQConnectionFactory("tcp://localhost:61616");
Connection connection;
  ..........}
```

这里将对客户端传递到服务端的消息进行不断的监听。

### 对客户端的消息进行转发：

```
public  void sendMsg(String msgText,String toipcName){
...............
}
```

这部分代码是对服务端将自己收到的消息按照“根据客户端状态判断”，将所收到消息转发到对应的其余客户端的功能实现。

## （三）测试代码部分：

### Client 端测试：

login()方法测试：

应用几组之前写入的“用户名+密码”进行登录，检验登录功能是否正常：

```
client.Login("liu","123");
client.Login("dfdsaf","123");
..............
client.Login("","");
```

SendMsg()方法测试：

分别在以下四种情况下对客户端发送消息功能方法进行测试：

```
//“用户名+密码”正确、用户名错误、密码错误、某秒请求次数>5
client.sendMsg("dsfadsa","username",true);
client.sendMsg("","username",true);
client.sendMsg("dadfafasdf","",true);
client.sendMsg("dafas","username",false);
client.sendMsg("","",true);
client.sendMsg("","",false);
```

### Server 端测试：

findUser()方法测试：

```
//用户名存在的情况下
Object user1 = testFindUser.invoke(server, "liu");
Assert.assertNotNull(user1);
//用户名不存在的情况下
```

```
Object user2 = testFindUser.invoke(server, "l");
Assert.assertNull(user2);
```

针对用户名存在或不存在这两种情况进行测试，在不同情况下反馈不同的结果。

checkConnection()方法测试：

此函数用以测试在三种情况下的客户端与服务端连接的稳定性，检验连接在各自情况是否皆能够正常维持：

```
//在登陆状态下测试
user.isLogin = true;
...........
//在未登录状态下测试
user.isLogin = false;
...........
//用错误的用户名测试
Boolean result3 = server.checkConnection("llll");
Assert.assertFalse(result3);
```

sendMessage()方法测试：

对于客户端所处的几种不同状态下进行测试，判断每种情况下发送信息的反馈是否符合要求：

```
//可以发送消息状态
user.sendMessagesNum = 80;
Boolean result1 = server.sendMessages("liu");
Assert.assertTrue(result1);
//不可以发送消息状态，即发送消息已超过了 100 条
user.sendMessagesNum = 180;
............
//用户名不存在的情况下
Boolean result3 = server.sendMessages("l");
Assert.assertFalse(result3);
```

checkTime()方法测试：

这个方法主要是测试根据 1s 内客户端发送消息数量对其进行限制功能

功能的实现与否：

```
//请求次数大于 0 并且距离上次请求时间小于 1s 的情况下的两种情况
     //<1>请求次数小于 5 次
    user.loginDate = System.currentTimeMillis();
    user.loginRequsetTime = 3;
    Boolean result2 = (Boolean) testCheckTime.invoke(server, user);
    Assert.assertTrue(result2);
    //<2>请求次数大于 5 次
    ..............
    user.loginRequsetTime = 6;
   ..............
    //请求次数不为 0 并且请求时间间隔大于 1s
    user.loginDate = System.currentTimeMillis()-2000;
    user.loginRequsetTime = 6;
    Boolean result4 = (Boolean) testCheckTime.invoke(server, user);
    Assert.assertTrue(result4);
```

login()方法测试：

主要对以下情况进行测试，反馈结果判断功能的实现情况；

```
    //能正常登陆的情况下
    int result1 = server.login("liu", "123");
    Assert.assertEquals(result1, 200);
    //密码不正确的情况下
    Int result2 = server.login("liu", "0");
    Assert.assertEquals(result2, 201);
    //每秒请求次数大于 5 次
    Method getFindUser = server.getClass().
                  getDeclaredMethod("findUser",String.class);
    getFindUser.setAccessible(true);
    User user = (User)getFindUser.invoke(server, "liu");
    ..............
    Assert.assertEquals(result3, 203);
    //用户名不存在
    int result4 = server.login("l", "123");
    Assert.assertEquals(result4, 202);
```

# 三．参考文献


- 《软件工程导论（第五版）》 张海藩著 清华大学出版社
- 《实用软件文档写作》 肖钢著 清华大学出版社

[ISBN：9787302103738]