《软件体系结构与设计》

第三章

作业4

班　　级：111171

学　　号：20171000970

学 生 姓 名：董安宁

指 导 教 师：尚建嘎

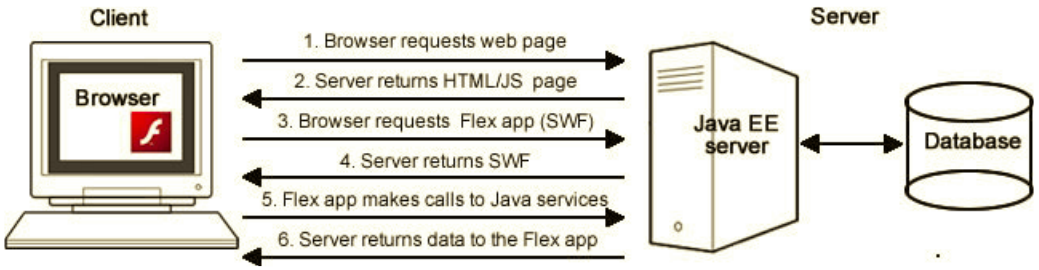
中国地质大学地理与信息工程学院软件工程系

2019 年 10 月

第 3 章 软件体系结构风格

作业 4

1. 传统的 B/S 架构有什么缺点？根本原因是什么？为改进这些缺点出现了哪些 RIA(富因特网应用程序)技术？下图所采用的是哪种 RIA 技术？并说明与传统 B/S 架构相比，这一技术有哪些改进之处。



缺点

① 输入客户端浏览器一般情况下以同步的请求/响应模式交换数据，每请求一次服务器就要刷新一次页面；

② 受 HTTP 协议“基于文本的数据交换”的限制，在数据查询等响应速度上，要远远低于 C/S 体系结构；

③ 数据提交一般以页面为单位，数据的动态交互性不强，不利于在线事务处理(OLTP)应用

④ 受限于 HTML 的表达能力，难以支持复杂 GUI (如报表等)。

根本原因

在于其架构体系所造成的约束和限制，以及现在网络浏览者更高的体验需求。

主要技术：

Adobe Flex

Flex 是一个高效、免费的开源框架，可用于构建具有表现力的 Web 应用程序，这些应用程序利用 Adobe Flash Player 和 Adobe AIR, 可以实现跨浏览器、桌面和操作系统

Microsoft silver light

一个跨浏览器的、跨平台的插件，为网络带来下一代基于.NETFramework 的媒体体验和丰富的交互式应用程序。

本图中采用的技术

Flex 技术

与传统 B/S 相比的优点

可以在任何地方进行操作而不用安装任何专门的软件。只要有一台能上网的电脑就能使用，客户端零维护。系统的扩展非常容易分层，易于管理和操作

2. **上机调试课堂上给出的 Java Chat Application 程序代码，并试着和你的同学/朋友使用这款软件进行聊天，你认为还有哪些功能需要修改完善？请给出你的实现方案。**

与同学处于同一个热点下，在同学的机器上运行客户端，可以获取到客户端 ip 地址，如下图所示

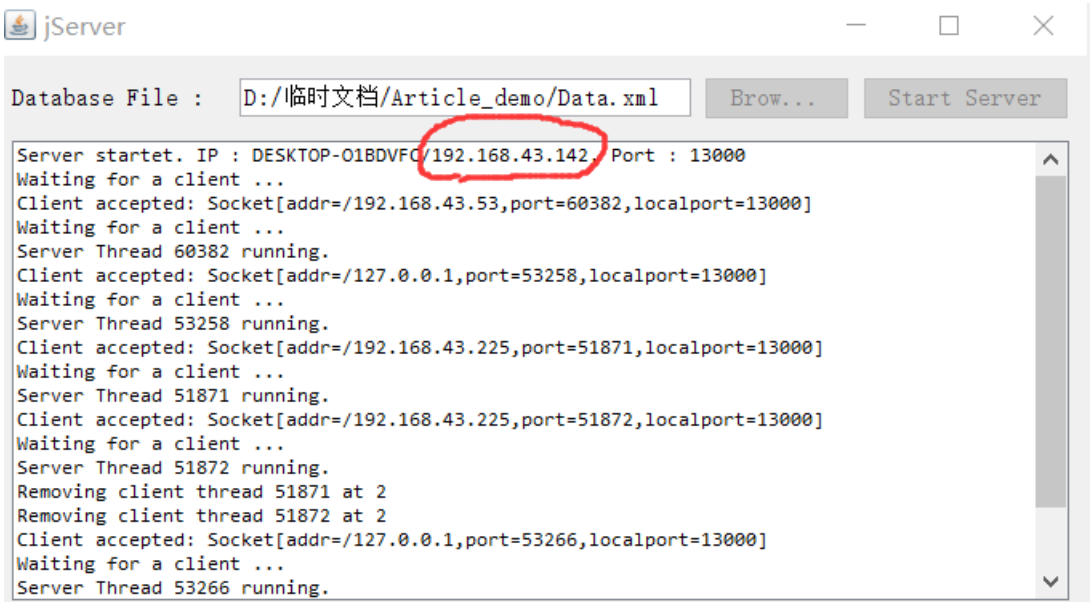


图 1 同学电脑上的客户端

在自己的电脑上运行 jMessenger，输入上面取到的 ip 地址，如下图所示，便可以登录成功，从右边可以看到和我一起连接上这个客户端的别的同学。

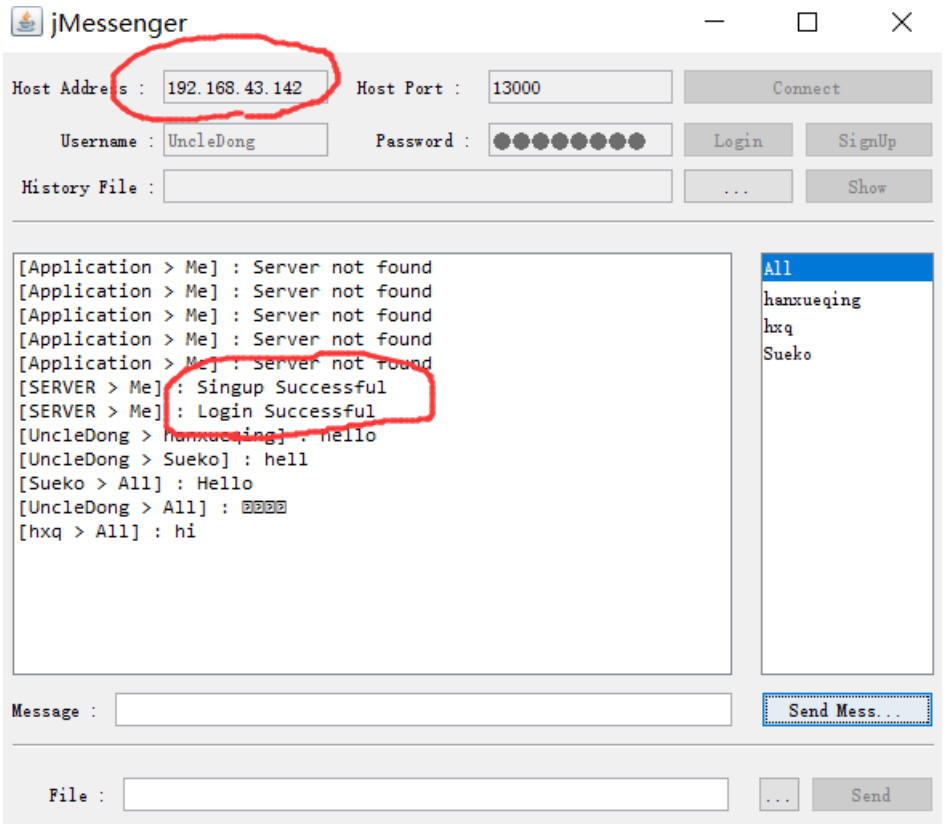


图 2 登录成功

还可以保存聊天内容并查找，如下图所示：

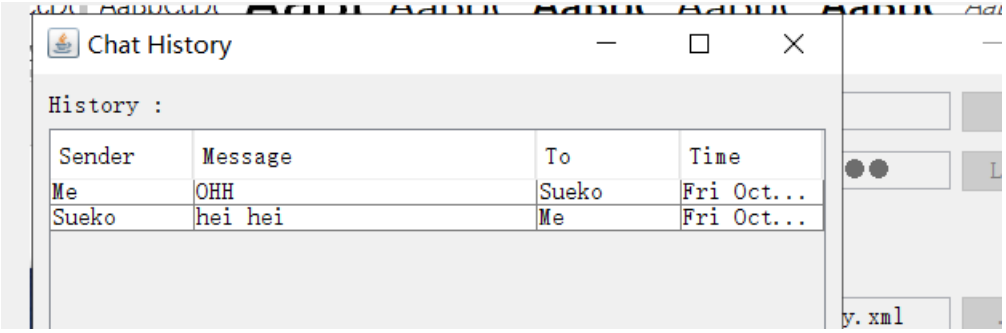


图 3 聊天记录

需要改善的功能：

① 不支持中文聊天，可以添加编码解码的功能，方法是在字符串发送和接收的时候进行处理。

② 可以添加多语言版本的界面

③ 消息接收都在一个文本框内，可以对不同用户分出不同的聊天框。方法是新建窗口，在接收到消息的时候分析是谁发给自己的，然后再据此显示到不同的窗口上。

3. 上机调试课堂上给出的 Java MVC 架构小程序，并试着将其改为 MVP 模式，分析 MVC 和 MVP 各自的优缺点和适用场合。

MVC 模式

课堂上的代码很以经将文件结构分成了 Model ，View，Controller 三个类，就分别对应着 MVC 三个单词，因此如果想实现课堂上的功能，只需要在 Controller 中添加视图就可以了。我初始化了该界面，并得到了下图的结果：

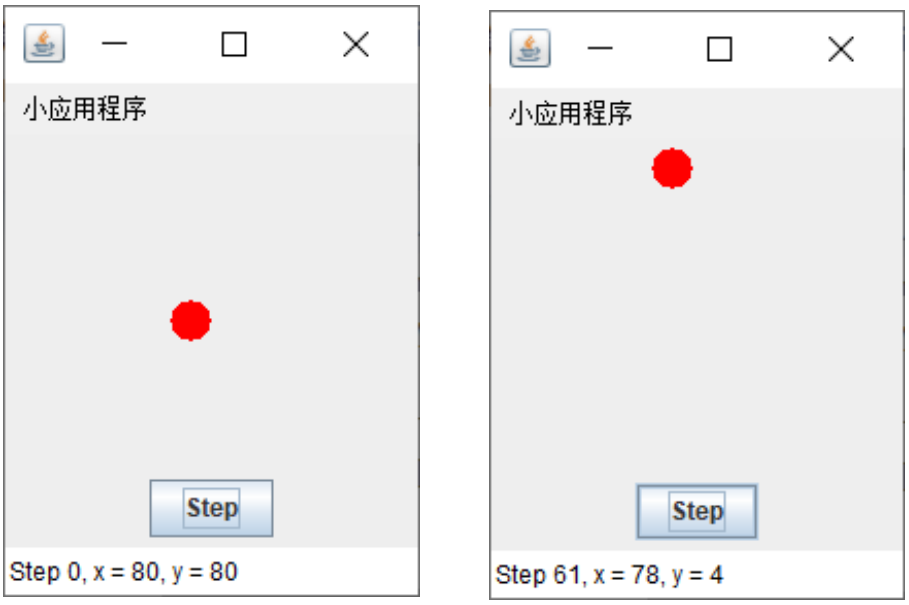


图 4 MVC 小程序应用

MVP 模式

新建 Presenter 类，功能和 Controller 类似，只不过更新视图和更新 Model 的功能都在 Presenter 类中实现，如图所示：

```
Model model = new Model();
//这里和Controller不一样，没有传入model
View view = new View();
public Presenter() {
    init() ;
    view.SetModel(model);
    start();
}
public void init() {
    setLayout(new BorderLayout());
    buttonPanel.add(stepButton);
    this.add(BorderLayout.SOUTH, buttonPanel);
    this.add(BorderLayout.CENTER, view);
stepButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent event) {
        //在这里更新模型
        model.makeOneStep();
        //在这里更新View
        view.SetModel(model);
    }});
    view.presenter=this;
}
```

图 5 修改后的类

可以看到 model 是在 Presenter 中动态更新的，每次按下按钮就会更新一次 model，并且 view 加载一次 model，这样就完成了 model 和 view 的更新。结果如右图所示，可以看到效果一样（除了在框框最下面没有坐标的表示了）

图 7 MVP 小应用程序

MVC 优缺点：

优点：

1：耦合性低

视图层和业务层分离，这样就允许更改视图层代码而不用重新编译模型和控制器代码，很容易改变应用程序的数据层和业务规则。

2：重用性高

只要接口相同，多个视图能共享一个模型，

3：部署快

使用 MVC 模式使开发时间得到相当大的缩减。

4：可维护性高

由于分层的原因，可以单独对需要修改的层进行修改，降低维护成本，和增加新的功能，提高代码重用性，从而提高开发效率。

5：有利软件工程化管理

由于不同的层各司其职，每一层不同的应用具有某些相同的特征，有利于通过工程化、工具化管理程序代码。

缺点

1：没有明确的定义

MVC 的定义各式各样，不同工程不尽相同的结构都可以解释成 MVC。同时由于模型和视图要严格的分离，这样也给调试应用程序带来了一定的困难。每个构件在使用之前都需要经过彻底的测试。

2：不适合小型，中等规模的应用程序

花费大量时间将 MVC 应用到规模并不是很大的应用程序通常会降低开发效率。

3：增加系统结构和实现的复杂性

不恰当的分层会增加结构的复杂性，并产生过多的更新操作，降低运行效率。

4：视图与控制器间的过于紧密的连接

视图与控制器是相互分离，但却是联系紧密的部件，妨碍了他们的独立重用。

5：视图对模型数据的低效率访问

依据模型操作接口的不同，视图可能需要多次调用才能获得足够的显示数据。对未变化数据的不必要的频繁访问，也将损害操作性能。

6：一般高级的界面工具或构造器不支持模式

改造这些工具以适应 MVC 需要和建立分离的部件的代价是很高的，会造成 MVC 使用的困难。

MVP 优缺点：

优点：

(1)降低耦合度
(2)模块职责划分明显
(3)利于测试驱动开发
(4)代码复用
(5)隐藏数据
(6)代码灵活性

缺点：

由于对视图的渲染放在了 Presenter 中，所以视图和 Presenter 的交互会过于频繁。如果 Presenter 过多地渲染了视图，往往会使得它与特定的视图的联系过于紧密。一旦视图需要变更，那么 Presenter 也需要变更了。

参考资料

[1]MVC 优缺点：https://blog.csdn.net/sunforraining/article/details/79015080
[2]MVP 优缺点：https://blog.csdn.net/jiuba5/article/details/79859353
[3]Java 聊天工具：
https://www.codeproject.com/Articles/524120/A-Java-Chat-Application