计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2017,26(9):87−92 [doi: 10.15888/j.cnki.csa.005973]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于 WebSocket 的消息推送系统①

王佃来[1], 宿爱霞[2], 安晏辉[1]

[1](首钢工学院 信息工程系, 北京 100144)
[2](中国软件评测中心, 北京 100048)

摘　要: 为解决学团管理工作中教师与学生信息交换效率低和反馈消息统计困难等问题, 基于 WebSocket 协议设计实现了一个包括教师端和学生端的实时消息推送系统. 并对系统设计实现中的关键模块和技术进行深入研究和分析. 测试结果表明, 该系统稳定可靠, 满足消息推送软件效率和实时方面的需求.

关键词: WebSocket 协议; 消息推送; 实时性

引用格式: 王佃来,宿爱霞,安晏辉.基于 WebSocket 的消息推送系统.计算机系统应用,2017,26(9):87–92. http://www.c-s-a.org.cn/1003-3254/5973.html

Message Pushing System Based on Websocket Protocol

WANG Dian-Lai[1], SU Ai-Xia[2], AN Yan-Hui[1]

[1](Department of Information Engineering, ShouGang Institute of Technology, Beijing 100144, China)
[2](China Software Testing Center, Beijing 100048, China)

Abstract: A real-time message pushing system, including teacher side and student side, is designed and implemented based on the WebSocket protocol aimed to improve the low efficiency in information exchange between teachers and students and to overcome the difficulty in getting the statistical feedback. At the same time, the key models and major technology are thoroughly investigated and analyzed. The results of the test indicate that the proposed system is stable and robust, and it meets the needs of message pushing software in terms of high efficiency and property in real-time.

Key words: WebSocket protocol; message pushing; real-time

随着高等院校学生规模的不断扩大和学生管理质量要求的不断提高, 负责学生管理工作的教师与学生之间的信息交换量大幅度提高, 打电话和发短信等传统沟通模式存在工作量大、效率低和数据无法统计分析等缺点. 因此, 迫切需要一种全新的、高效的信息传递工具或方式, 解决学生与学生管理教师之间的信息交换问题.

近年来信息技术飞速发展, 人类对移动通讯和网络信息的需求急剧上升. 迄今为止, 全球移动通讯用户已超过 50 亿, 互联网用户已合逾 20 亿, 中国移动通讯用户达到 7.38 亿, 移动通讯和互联网已成为当代人不可或缺的信息来源和交流工具. 而大学生则是其中最具活力的人群, 基本上人手一部智能手机, 并且均接入互联网, 手机成为当代大学生信息获取和交流沟通必备工具. 这为消息推送系统的设计与实现提供了坚实的物质基础, 也为解决学生与教师之间高效信息交换难题提供了契机.

移动通讯技术和互联网技术的高速发展也带动了消息推送和即时通讯系统的蓬勃发展. 目前, 移动推送技术的实现主要有以下几种方式: 基于移动平台内置的推送技术、第三方推送平台技术和基于开放的网络协议实现的推送技术(如基于 XMPP, MQTT 或 WebSocket 等), 其中基于开放网络协议的推送技术具有良好的适用性和灵活性, 受到许多开发的亲睐. 关于

① 收稿时间: 2016-12-30; 采用时间: 2017-01-16

XMPP、MQTT 和 WebSocket 协议的详细介绍和优缺点比较请参阅文献[10], 在此不再赘述.

本文要解决的首要问题是方便快捷的实现学生和教师之间的消息传递, 更确切的说是如何方便快速的将学生管理类信息如: 学团管理、班级管理、招生就业和毕业信息传达给学生, 并且确认学生是否成功接收. 因此, 教师与推送服务器端的接入软件最好是不需要任何安装就可以使用的软件, 如浏览器, 而且教师发送软件时一般会使用操作方便的电脑, 这样可提高录入信息和发送信息效率; 学生端接入服务器方式应该不受时间和空间的限制, 因此手机 App 软件恰好解决了这个问题, 学生一般情况无法一直在电脑前, 但其手机会随身携带. 通过上述分析可以看出教师与服务器之前的协议最好支持 Web, 因此 WebSocket 协议成为本文开发的首选. WebSocket 是 HTML5 提出的一个新的通信协议, 它可以在服务器和浏览器之间架设一个全双工的通信信道并使用套接字来传输数据[8]. WebSocket 协议主要用于解决基于 Web 的推送, 实时性强, 效率高. 有许多学者基于 WebSocket 协议实现了相关的消息推送系统, 请参阅文献[5-7,11].

基于上述分析, 本文在研究了 WebSocket 协议的基础上设计并实现了一个消息推送系统, 该系统解决了教师与学生之间的消息推送问题, 并在学院信息工程系做了测试和试运用.

1 消息推送系统的设计

1.1 系统总体结构设计

消息推送系统主要包括以下模块: 教师消息推送子系统、WebSocket 消息服务子系统和学生消息接收子系统, 系统结构图如图 1 所示.

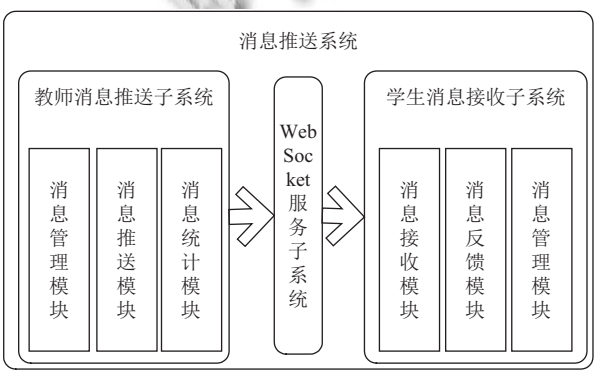

图 1 消息推送系统结构图

教师消息推送系统包括消息管理模块、消息推送模块和消息统计分析模块. 消息管理模块完成消息的添加、修改、删除、查询和审核功能; 消息推送模块将通过审核的消息推送学生的功能; 消息统计分析模块完成对推送消息的成功率、学生回馈信息和消息月、周统计等功能.

WebSocket 消息服务子系统主要完成教师和学生之间消息的实时传功能. 为了实现实时性, 采用广播机制, 将收到的消息广播给每一个连接到 WebSocket 服务的用户.

学生消息接收子系统包括消息接收模块、消息反馈模块和消息管理模块, 主要完成消息接收、消息反馈和手机 App 端消息显示、查询与删除功能.

1.2 系统网络结构设计

系统网络结构如图 2 所示. 为保证消息的准确性避免错误和无效消息的不良影响, 教师用户分为以下三种角色: 消息编辑员、消息审核员和消息发送员. 首先, 消息编辑员通过 Internet 使用部署在 Web 服务器端的教师消息推送子系统完成消息的编辑和修改, 确认消息无误后交由消息审核员审核, 消息审核无误后, 消息发送员与 WebSocket 消息服务连接并通过该服务完成消息发送; 如果消息未通过审核则消息返回消息编辑员重新编辑, 直到消息无误后再提交上述流程进行消息发送. 发送后的消息通过消息统计模块分析消息发送的情况, 根据消息接收的成功率决定是不是对接收失败的消息重新发送.

WebSocket 消息服务端完成消息的发送和接收中转, 实现将教师下发的消息推送到学生端, 此外完成学生端的反馈消息返回到教师端.

学生用户通过手机 APP 经由无线网络与 WebSocket 消息服务连接, 完成消息的接收, 并按消息要求完成反馈任务.

1.3 系统关键模块设计

系统中主要涉及教师端消息推送、WebSocket 消息服务和学生端消息接收三个关键模块, 下面分别对其进行详细描述.

(1) 教师端消息推送模块设计

为了方便操作和便用, 教师端消息推送模块使用网页模式实现消息推送. 首先, 网页的 onload 事件中判断浏览器是否支持 WebSocket, 如果不支持在页面中给出提示, 建议用户更换支持 WebSocket 的浏览器. 如

果支持则尝试与服务器建立连接, 如果建立连接成功给出提示, 可进行消息推送, 如果建立连接不成功, 提示用户无法建立连接, 消息无法推送. 具体流程如图 3 所示.

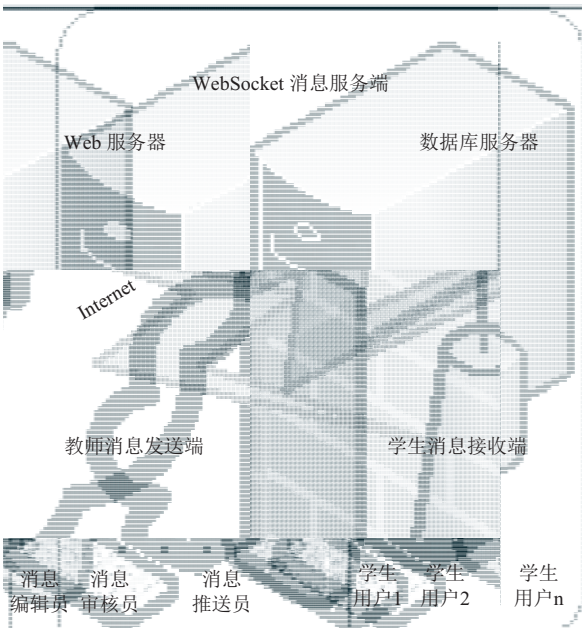

图 2 系统网络结构图

(2) WebSocket 服务模块设计

WebSocket 服务模块主要完成以下功能: 教师端和学生端用户的连接维护、将教师端的消息推送到学生端用户和处理学生的反馈信息.

教师端和学生端用户的连接维护功能采取以下机制实现: 首先, 为保证系统安全, 在建立连接时需要用户鉴权, 鉴权成功的用户, 将以一个唯一标识加入消息推送链表, 该链表做为推送消息的用户列表, 在 WebSocket 服务内存中维护, 鉴权失败的用户将做抛弃处理. 其次, 鉴权成功用户的唯一标识同时被写入数据库用做消息推送数据对比. 教师端由于用户较少, 唯一标识采用手工分配的方式, 学生端则采用由手机 App 自动生成的方式. 最后, 如果用户退出连接, 则将该用户的唯一标识从消息推送链表中移除, 数据库中并不删除该用户信息. 连接建立流程见图 4.

消息推送功能采取以下方式实现: 首先, 读取 WebSocket 服务端的消息推送链表数据(记为 listWebSocket)和数据库中的用户列表(记为 listDB); 然后从 listDB 中的取一用户, 判断该用户是否在 listWebSocket 中, 如果在则直接将消息推送给用户, 否则将消息写入未接收消息数据表, 表示该用户没有接收到该消息, 用户在下次登录时将发送给用户, 如果用户未接收消息超过 10 条, 则只发最近的 10 条消息, 避免用户手机端接收过多消息. 依次循环直到 listDB 中的所有用户都被遍历为止.

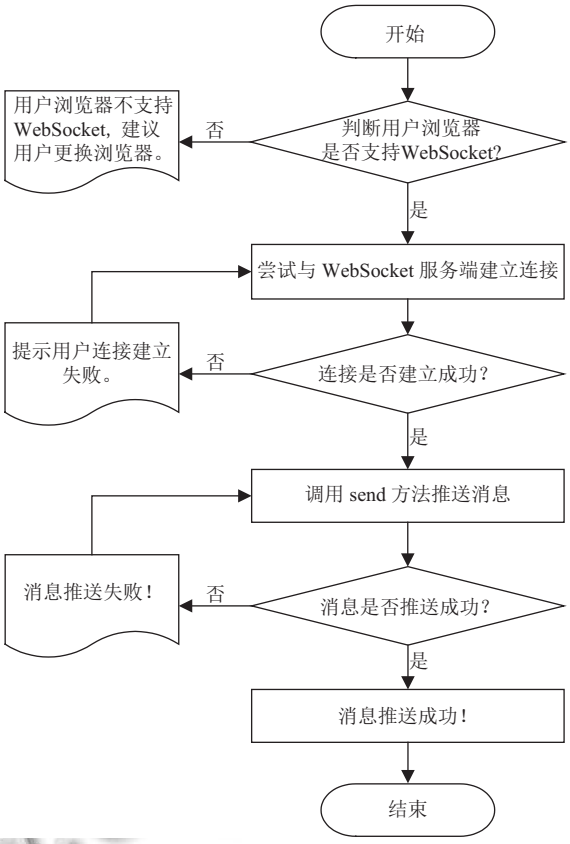

图 3 教师端消息推送流程图

(3) 学生端消息接收模块设计

学生端消息 App 采取以下流程实现消息接收. 首先, 手机 App 在移动端生成一个唯一的 UUID 和用户鉴权加密字符串, 以上述两个字符串为参数与 WebSocket 服务发起建立连接请求, 服务端使用加密字符串进行鉴权, 鉴权成功则成功连接建立. 成功建立连接后, 在手机 App 端实现 WebSocket 协议监听连接状态, 主要监听以下三个方法: onMessage()、onError()和 onClose()方法. 如果 onMessage()方法中接收到消息, 则调用手机端的 Handler 完成振铃、将消息写入到手机端数据库等操作; 如果 onClose()方法调用, 则说明连接关闭, 手机 App 弹出提示信息通知用户连接关闭, 并尝试请

重新连接; 如果 onError()方法调用, 则说明发生错误, 将弹出提示信息通知用户, 并尝试重新启动 WebSocket 连接建立模块.

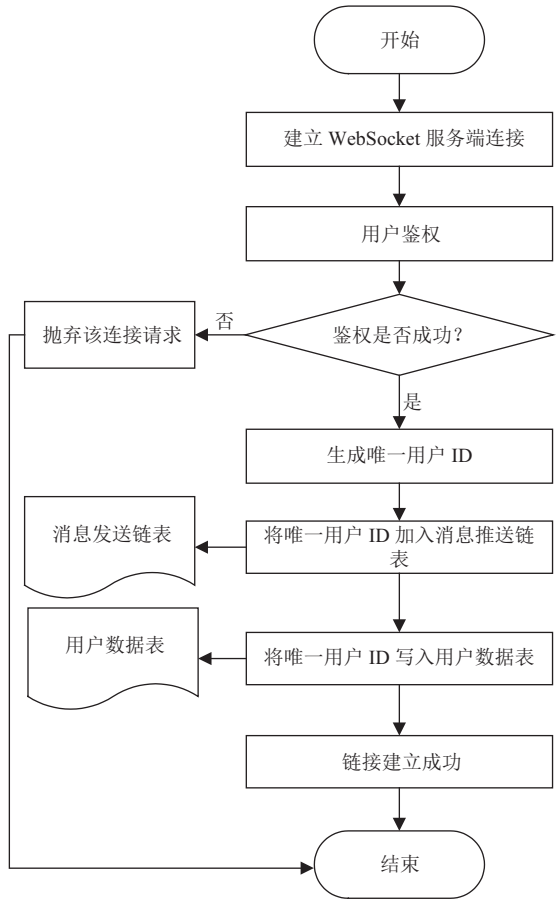

图 4 用户与 WebSocket 服务建立连接流程

2 消息推送系统的实现

2.1 系统开发环境

消息推送系统的开发环境如下所示: 操作系统为 window8.1, 开发工具为 Eclipse4.5.1, JDK1.7.X, Web 服务器采用 Tomcat8.x;WebSocket 的服务端实现使用 Tomcat8.x 的 API 进行二次开发而成, WebSocket 移动端的实现采用开源 http://java-websocket.org/提供的客户端协议实现, 并做相应修改完成, 浏览器使用 Google Chrome, 版本 48.0.2564.10 m (64-bit).

2.2 系统关键模块实现

(1) 教师端消息推送模块实现

消息推送系统教师端的设计采用 Struts2、Spring 和 JPA 集成框架开发, 其中 Struts2 主要为控制层, Spring 为业务辑层的 Java Bean 提供整体的管理功能, JPA 提供数据库的操作功能, 整体采用注解方式实现, 很好的减少了繁琐的配置文件配置, 提高了开发效率, 教师端实现界面如图 5 所示.

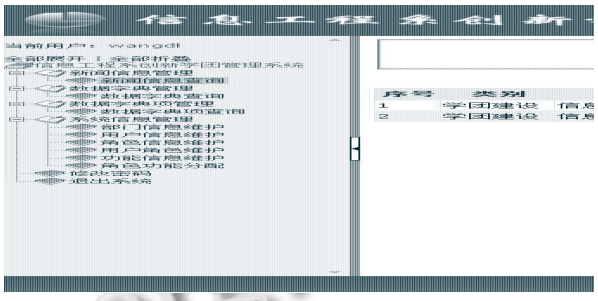

图 5 消息推送端用户界面

通过浏览器推送消息的核心代码实现如下:

```
<script language="text/javascript">
   //创建 WebSocket 对象
var url="ws://localhost/MeWebSocket";
var ws = new WebSocket(url);
//Websocket 服务连接打开监听方法
ws.onopen = function(){ }
//WebSocket 服务调用信息接收监听方法
ws.onmessage = function(){ }
//WebSocket 服务调用连接关闭监听方法
ws.onclose = function(){ }
//WebSocket 服务出错监听方法
ws.onerror = function(){ }
//发送消息
function sendmsg(message){
   if(ws!=null&& message!=''){
      ws.send(message);
      alert("消息已发送！ ");
   }
}
</script>
```

(2) WebSocket 消息服务模块实现

WebSocket 有许多 Web 服务器提供了实现, 本文基于 Oracle 发布的 JSR356 规范, 使用 Apache Tomcat8.x 下的实现进行二次开发而成. 其核心代码如下:

```
@ServerEndPoint(value="/MeWebSocket")
public class MeWebSocket{
```

```
@OnMessage
public void onMessage(String msg, Session session)throws Exception{
  //处理客户端消息, 可向客户发送消息
  session.getBasicRemote().sendText(msg);
}
@OnOpen
public void onOpen(){
  //建立并打开连接时要进行的处理
}
@OnClose
public void onClose(){
  //关闭连接时进行的处理
}
@OnError
public void onError(Throwable t){
  //出错时要进行的处理
}
```

```
}
```

其中, @ServerEndpoint 是一个类层次注解, 它实现将当前类定义成一个 WebSocket 服务器端, value 指定用户连接访问的服务器 URL 地址. @OnOpen、@OnClose 和@OnError 注解分别指定当 WebSocket 服务打开、关闭和出错时执行的方法. @OnMessage 注解标明当服务器接收到客户端消息时执行 onMessage()方法.

(3) 学生端消息接收模块实现

学生端手机 App 基于 android4.3 平台开发, android 端 WebSocket 协议使用 http://java-websocket.org/提供的 jar 包, 并其基础上进行封装. 实现了与 WebSocket 服务端边接功能. 具体实现过程如下: 首先, 手机端 App 启动时在 Activity 的 onCreate()方法中完成与 WebSocket 服务器的连接; 建立成功能后监听 WebSocket 的以下事件 OnMessage()、OnClose()和 OnError()事件, 完成消息的接收、连接关闭和出错信息. 具体类图见图 6.

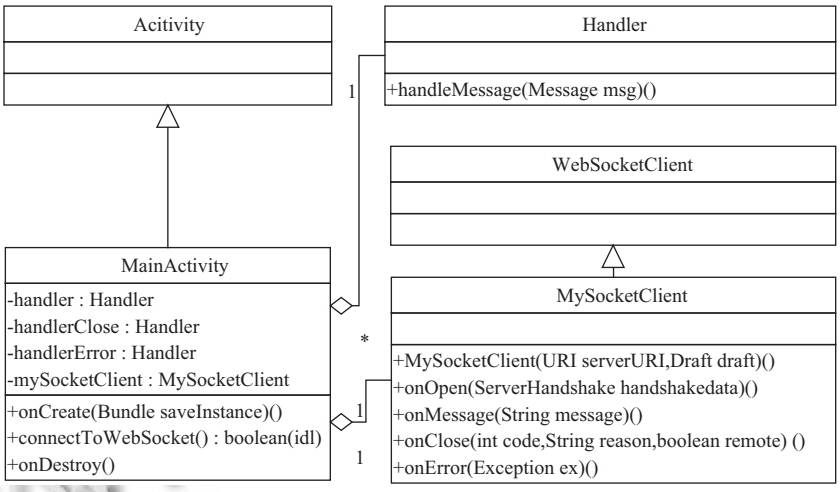

图 6 学生端消息发送类图

图 6 为手机端主要类的类图, 图中 MainActivity 类是手机 App 的启动界面, 在 onCreate()方法中调用 connectToWebSocket()方法完成与服务器端的连接, 该类有四个主要的成员变量: handler、handlerClose、handlerError 和 mySocketClient, 其中 handler、handlerClose 和 handlerError 重载 Handler 类的 handleMessage 方法分别处理 MySocketClient 监听的 onMessage()、onClose()和 onMessage()方法. MySocketClient 类主要完成与 WebSocket 服务器的连接与消息事件的监听与响应, 其 onMessage()负责处理 WebSocket 服务器端消息, onClose()方法处理与 WebSocket 连接的关闭事件, onError()方法处理 WebSocket 错误事件. 基于效率和程序健壮性的考虑 MySocketClient 的上述三个方法均采取多线程机制, 更好完成消息处理. 同时, 为了提高效率和开发方便性, MySocketClient 作为 MainActivity 的内部类, 这种机制

可在 MySocketClient 的三个方法中直接使用 handler、handleClose 和 handleError 三个成员变量.

3 结语

本文基于 WebSocket 协议实现了一个实时消息推送系统, 系统实现后在学院信息工程系师生中进行了测试和试运行, 结果表明该系统运行稳定, 有较高的可扩展和应用价值. 在该系统的帮助下, 学生管理工作教师与学生之间的信息交换效率和质量有较大提高, 在一定程度上提升了信息工程系统的信息化水平, 促进了学团管理工作水平和质量的提高.

目前, 系统只实现了 Android 手机端软件, 对苹果手机还没有开发相应的学生端软件, 下一步的工作是完成 iOS 学生端系统软件的设计与开发. 此外, 值得注意的是本文采用的 WebSocket 协议是一种正在发展的 Web 规范, 存在一定的兼容性和扩展问题, 且实现是在 Tomcat8.x 提供的 API 基础上进行封装, 也有一定的局限性, 下一步工作重点是基于开源 WebSocket 项目独立实现该协议, 进一步完善消息推送系统.

参考文献

1 孙清国, 朱玮, 刘华军, 等. Web 应用中的服务器推送技术研究综述. 计算机系统应用, 2008, 17(11): 116–120. [doi: 10.3969/j.issn.1003-3254.2008.11.030]
2 彭亮. 面向移动设备的 XMPP 协议的研究与应用[硕士学位论文]. 长沙: 中南大学, 2014.
3 王美妮, 王颖, 赵伟. 基于 XMPP 协议消息推送机制的研究与实现. 长春师范学院学报(自然科学版), 2014, 33(1): 27–31.
4 Pimentel V, Nickerson BG. Communicating and displaying real-time data with WebSocket. IEEE Internet Computing, 2012, 16(4): 45–53. [doi: 10.1109/MIC.2012.64]
5 张玲, 张翠肖. WebSocket 服务器推送技术的研究. 河北省科学院学报, 2014, 31(2): 49–53.
6 李锡辉, 杨丽. 基于 WebSocket 的服务器推送技术研究. 网络安全技术与应用, 2014, (6): 45–46.
7 张艺. 基于 WebSocket 的即时通信系统研究与实现. 软件, 2015, 36(3): 89–94.
8 陆晨, 冯向阳, 苏厚勤. HTML5 WebSocket 握手协议的研究与实现. 计算机应用与软件, 2015, 32(1): 128–131, 178.
9 杨健, 周渊平. 基于 Android 平台的多人视频聊天系统. 计算机系统应用, 2016, 25(1): 75–79.
10 陈涛, 李娟. 基于 MQTT 协议的推送技术研究. 软件导刊, 2016, 15(3): 18–21.
11 陈炜, 苏厚勤, 柴炯. 基于 WebSocket 技术水文资源监管系统的研究与实现. 计算机应用与软件, 2016, 33(3): 104–108, 113.