基于Android的实时共享白板的设计与实现

目录

[摘 要： 3](#_Toc482658823)

[Abstract: 4](#_Toc482658824)

[1. 绪论 5](#_Toc482658825)

[1.1. 背景 5](#_Toc482658826)

[1.2. 意义 5](#_Toc482658827)

[1.3. 目标 5](#_Toc482658828)

[1.4. 可行性分析 6](#_Toc482658829)

[1.5. 相关技术说明 6](#_Toc482658830)

[1.6. 软硬件环境 9](#_Toc482658831)

[2. 需求分析 11](#_Toc482658832)

[2.1. 功能需求 11](#_Toc482658833)

[2.2. 性能需求 15](#_Toc482658834)

[2.3. 质量属性 15](#_Toc482658835)

[2.4. 对外接口 16](#_Toc482658836)

[2.5. 约束 16](#_Toc482658837)

[3. 概要设计 17](#_Toc482658838)

[3.1. 网络结构设计 17](#_Toc482658839)

[3.2. 功能结构设计 18](#_Toc482658840)

[3.3. 数据库设计 19](#_Toc482658841)

[3.4. 综合考虑 23](#_Toc482658842)

[4. 详细设计 24](#_Toc482658843)

[4.1. 模块设计 24](#_Toc482658844)

[4.2. 接口设计 24](#_Toc482658845)

[结论 25](#_Toc482658846)

[致谢 26](#_Toc482658847)

[【参考文献】 27](#_Toc482658848)

[附录： 27](#_Toc482658849)

摘 要：随着互联网的飞速发展，各种出行类APP和网站也层出不穷，“晥美出行”则是一个主要提供网上车次查询、网上订票、网上购票、网上退票等客运汽车票的服务的网站平台。也由此“晥美出行”需要和客运站系统进行对接，获取客运站数据并且进行相关操作。由于晥美出行平台不能直接对客运站系统进行任何更改，只能在允许的范围进行相关数据操作。所以基于此，本文研究了国内外WebService和SOA技术的发展情况，发现SOA这种粗粒度、松耦合的服务架构正好适合这个应用场景，设计并据此设计实现了一套基于WebService和SOA技术的Web接口系统。

关键词：WebService、接口系统、SOA

Abstract: English English English English English English English English English English English English English English English English English English English English English English English English

Keywords: English english English

1. 绪论
   1. 背景

随着我国普遍进入互联网时代，“互联网+”已经深入到国民工作生活的方方面面，人们逐渐习惯使用智能手机上的APP解决问题，譬如购物软件、外卖订餐软件、同城租房软件、打车软件等。随后，基于互联网，出现了另一个概念“共享经济”，指的是一种共用人力与资源的社会运作方式，它包括不同个人与组织对商品和服务的创造、生成、分配、交易、和消费的共享，源于“共享经济”的应用也比比皆是，譬如Uber、摩拜单车等。互联网以其“连接万物”的能力，高效地传递信息，同时结合分布式计算，可以带来超大的计算能力；“共享”很好地解决了社会资源不足的问题，提高了社会的整体资源利用率。这两点在某种程度上揭示了人们的需求，更简单快捷地解决以往需要现场沟通并支付资源独享费用的问题。

* 1. 意义

随着社会的飞速发展，人们的要求和期望越来越高，越来越多的行业参与到 ”互联网+” 的大家庭中，公司规模在变大，经常有多个公司并分布在不同的地方，项目的复杂度也在不断提升，项目组成员也随之不断增多。如此一来，沟通变得越来越重要，但也越来越难，尤其是跨地区沟通的效率低而成本高。如何解决这个问题? 网络给出了答案，因为高速网络联通了你我，使得我们可以跨越地理相隔实现交互，而传统的即时通讯工具一般局限于富文本聊天，文本相对于图像，缺少直观性，不能保证沟通的简单便捷高效。而基于Android的实时共享白板是通过创建一块虚拟会议空间，多人连入并共享这块空间，在里面可以通过白板绘画、发文字、发图片、发语音等方式实现信息的交互共享，从而简单快捷地解决团队沟通问题。

* 1. 目标

按照软件工程的方法，设计并实现一个基于Android的实时共享白板APP，要求它有较为完善的基础功能（例如注册、登录、重置密码）和具有特色的专职功能（例如共享白板、会议管理、联系人管理），同时保证软件的可靠性、可维护性，努力提升软件的易用性、兼容性、效率。

* 1. 可行性分析
     1. **难点及解决方案**
        1. **绘图控件的渲染**

难点：目标实现的白板，包括多种绘画形式，并且考虑到需求变更的可能，需要尽可能让各个绘画工具相互独立，同时为了提高用户体验，需要尽量提高画板的性能，从而让用户更流畅的操作。

解决：经过查资料，了解到使用Java的Canvas或HTML5的Canvas都可以实现，如果使用HTML5的Canvas，可以结合当下流行的React，而React的内部机制使得它相比原生JavaScript拥有更高的效率，可以满足高性能的需要，同时其组件化的思想可以满足易扩展易重构的需要。因此HTML5的Canvas与 React相结合可以解决问题。

* + - 1. **多人实时同步通信**

难点：目标实现的共享白板，需要实时同步信息到多个用户，这就需要服务器可以主动向客户端发送消息。

解决：经过查资料，HTTP协议具有被动性，虽然通过轮询的方式也可以实现实时通信，但是性能不佳，相比而言，HTML5的WebSocket协议可以更好地满足需求，建立WebSocket连接后，服务器可以主动向客户端发送消息。

* + 1. **分析结果**

综上，两个难点均可以得到有效解决，结合课题的预期难度、工作量以及自身情况得出，课题是可行的。

* 1. 相关技术说明
     1. **Java**

Java是一种面向对象的编程语言，它和c以及c++比较接近，但是在某些特性上，对前面两种语言进行了简化，譬如不再使用指针、多继承等等；它支持网络应用开发，提供了网络编程接口和相关的类库；它支持多线程，可以通过实现Runnable接口或者继承Thread类并重写run方法来创建一个线程对象；它支持动态载入，可以通过网络远程载入需要的类；它是解释执行的，先通过与平台无关的编译器把源程序.java编译成与平台无关中间码.class，然后根据目标运行系统选择不同平台的解释器，也就是说解释器是平台相关的，用相应的解释器把中间码.class解释成和平台相关的机器码，然后运行，从而实现一次编译到处运行的目标，也就是跨平台，如图1.1。

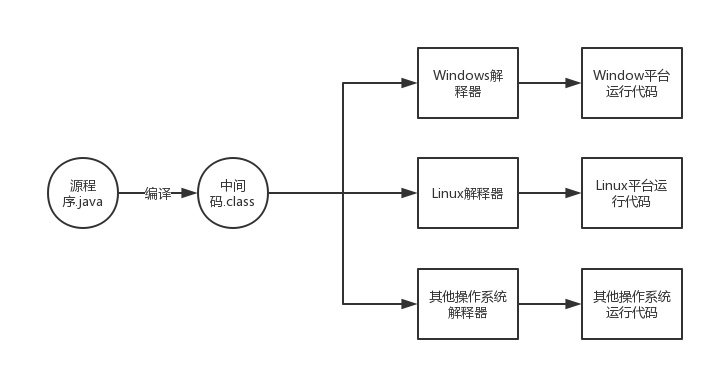


图1.1 Java编译解释过程

* + 1. **PHP**

PHP是一种通用脚本语言，主要用于生成动态网页内容、操作服务器上的额文件和数据库、数据加密、用户授权访问等。PHP凭借其开源、免费、易学、服务器兼容性好、跨平台等特性，被称为“世界上最好的语言”。PHP拥有超过150个扩展库，可供开发人员调用，很大程度上提高了开发效率和代码的性能，例如PDO扩展，它提供了PDO类来对数据库进行访问，开发人员使用不同数据库时候调用的函数名是相同的，使得应用层不用去关心具体要连接的数据库服务器的类型，除此之外，相比原生MYSQL，可以有效防御SQL注入攻击。

* + 1. **JavaScript**

JavaScript是一种解释型脚本语言，它是弱类型、函数优先的，一般作为开发网页的脚本语言，在其他环境也有使用。随着当前网站开发前后端更严格的分离，JavaScript负责越来越多的业务逻辑检查、处理等工作，而不仅仅是单纯的DOM操作和实现网站特效。

* + 1. **React**

React是Facebook研发的JavaScript库，具有易入门、代码易重构、更好支持响应式网页、高性能等特点。React的主要原理包括Virtual DOM、Components、State和Render。其中Virtual DOM就是在真实DOM上面抽象出一个对象，用来表示DOM应该怎么呈现，当需要更新页面的时候，不是直接更新真实DOM，而是更新Virtual DOM，React会等到当前事件循环结束，通过diff算法计算Virual DOM和真实DOM的差别，并计算出最小步数来进行更新，从而提高了性能；Virual DOM的每一个节点就是一个commponent，它的存在使得diff算法更高效；State包含该定义commponents的数据，当数据改变时，它会调用Render进行重新渲染。

* + 1. **WebSocket**

WebSocket是HTML5中的一个新的协议，它和HTTP协议基本没关系，HTTP中每个response都对应一个request，也就是只有客户端主动请求服务器才会回复，而不能做到服务器主动推送消息给客户端，基于这个原理的AJAX轮询和polling技术，可以变向实现实时交互，但是性能不佳且代价比较大；而WebSocket只需要客户端一次request，建立持久连接后，服务器便可以主动向该客户端推送任意多次消息，同时较AJAX轮询和polling技术具有更高的效率和性能。WebSocket.org 对传统的轮询方式和 WebSocket 调用方式作过一个详细的测试和比较，将一个简单的 Web 应用分别用轮询方式和 WebSocket 方式来实现，并把测试结果用柱状图进行表示，如图1.2。



图1.2 轮询和WebSocket的性能对比

* + 1. **Gateway Worker**

Gateway Worker是一个开源的Socket服务器框架，用PHP编写，该框架采用的是Gateway和Worker进程模型，它支持分布式部署、支持高并发、支持多种应用层、支持代码热更新、支持HVVM协议、提供心跳检测、在Linux服务器上可以做到守护进程化。使用该框架，开发者只需要简单配置并通过实现框架内部Events类的函数，就可以轻松监听WebSocket连接的建立、消息请求、连接断开等事件，并实现消息的单发、群发、广播、定时推送等功能。

* 1. 软硬件环境
     1. **硬件环境**

联想Y500笔记本:

CPU Intel Core i5-3230M 2.6GHz

RAM 8G

腾讯云服务器：

CPU 1核

RAM 2G

带宽 2M

小米4手机：

CPU: 高通 骁龙801（MSM8974AC）

RAM 3G

* + 1. **软件环境**

Windows8.1系统、Ubuntu14.04系统、Android 6.0系统、AndroidStudio2.2.3、 PhpStorm 2016.12、JDK 1.8、PHP 5.5.9、Apache 2.4.7、MySQL 5.5、MySQLWorkbench 6.3、SQLiteStudio 3.1.1、FiliZilla 3.25

1. 需求分析
   1. 功能需求
      1. **列举功能需求**
2. 用户可以注册
3. 用户可以登录
4. 用户可以注销
5. 用户可以重置密码
6. 用户可以更改头像
7. 用户可以更改姓名
8. 用户可以更改密码
9. 用户可以更改会议偏好
10. 用户可以检查更新软件
11. 用户可以反馈建议
12. 用户可以添加联系人
13. 用户可以接受添加联系人申请
14. 用户可以拒绝添加联系人申请
15. 用户可以删除联系人
16. 用户可以以主持人身份安排会议
17. 用户可以以主持人身份查看已安排的会议
18. 用户可以以主持人身份删除已安排的会议
19. 用户可以以主持人身份进入已安排的会议
20. 用户可以以主持人身份直接召开会议
21. 用户可以以主持人身份共享（图片或网页）
22. 用户可以以主持人身份结束共享（图片或网页）
23. 用户可以以主持人身份锁定会议
24. 用户可以以主持人身份取消锁定会议
25. 用户可以以主持人身份查看会议成员列表
26. 用户可以以主持人身份禁止与会者讲话
27. 用户可以以主持人身份取消禁止与会者讲话
28. 用户可以以主持人身份禁止与会者使用白板
29. 用户可以以主持人身份取消禁止与使用白板
30. 用户可以以主持人身份把与会者踢出会议
31. 用户可以以与会者身份加入会议
32. 用户可以以与会者身份观看主持人的共享
33. 用户可以以与会者身份查看会议成员列表
34. 用户可以邀请别人加会
35. 用户在会议中可以使用白板绘画
36. 用户在会议中可以给白板录屏
37. 用户在会议中可以进行群聊
38. 用户可以离开会议
    * 1. **建立用例模型**
         1. **识别参与者**

系统的参与者就是软件的注册用户，而注册用户可能扮演两种角色：主持人和与会者。故系统有三个参与者，分别是Registrant（注册用户）、Host（主持人）、Participant（与会者），其中注册用户和主持人之间是类属关系，注册用户和与会者之间也是类属关系。

* + - 1. **识别用例**

1. Register(注册)：提供用户新建个人账号的功能
2. Login(登录)：提供用户进入个人账号的功能
3. Logout（注销）：提供用户退出当前账号的功能
4. ResetPassword（重置密码）：提供用户忘记密码后重置密码的功能。
5. SetMyInfo（设置个人信息）：提供用户更改姓名、头像、密码的功能。
6. SetMeetingPrefs（设置会议偏好）：提供用户更改会议默认偏好设置的功能。
7. CheckUpgrade（检查更新）：提供用户获取新版本软件更新的功能。
8. FeedBack（反馈）：提供用户反馈使用情况、建议的功能。
9. ContactsManage（联系人管理）：提供添加、同意、拒绝、查询、删除联系人的功能。
10. ArrangeMeetingManage（会议安排管理）：提供会议安排、查看、删除。
11. HostMeeting（召开会议）：提供主持人召开会议的功能。
12. InviteOutter（会外邀请）：提供主持人会外邀请别人加会的功能。
13. InviteInner（会内邀请）：提供用户会内邀请别人加会的功能。
14. JoinMeeting（加入会议）：提供与会者加入会议的功能。
15. LeaveMeeting（离开会议）：提供用户离开会议的功能。
16. ShareRes（共享资源）：提供主持人会内共享（图片或网页）给与会者以及取消共享的功能。
17. LockMeeting（锁定会议）：提供主持人锁定会议、解锁会议的功能。
18. MeetingPrivilegeManage（会内权限管理）：提供主持人对与会者禁言、取消禁言、禁画、取消禁画、踢出会议的功能。
19. Board（白板）：提供用户会内使用白板的功能。
20. Chat（群聊）：提供会内群聊的功能。
21. Member（参与者）：提供查看会议参与者的功能。
    * + 1. **用例图**

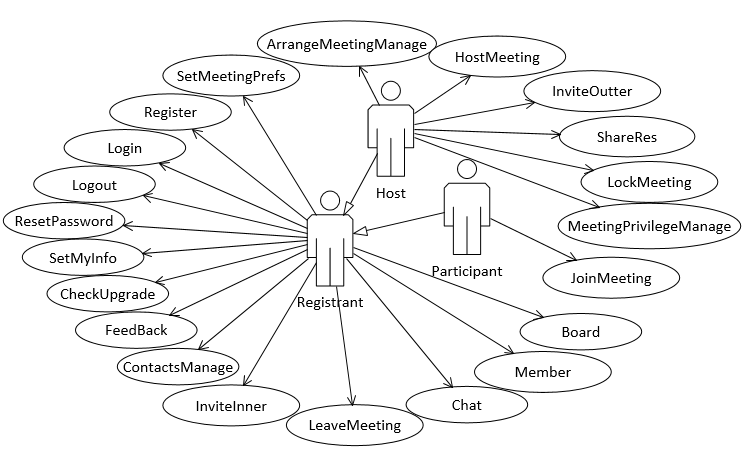


图2.1 系统用例图

* + 1. **业务流程分析**



图2.2 系统业务流程图

* 1. 性能需求
     1. **速度**

1. 注册验证码必须在15秒内发送到用户注册所用邮箱。
2. 登录必须在5秒内完成。
3. 所有用户查询必须在5秒内完成。
4. 开会、加会必须在10秒内完成。
   * 1. **负载**
5. 系统至少可以维持300个WebSocket并发连接。
6. 系统至少可以维持20个容量为20人的会议。
   * 1. **实时性**
7. 白板绘画、群聊信息必须在1.5秒内同步到会议的其他参与者。
8. 联系人的添加、接受、拒绝、删除、加会邀请必须在2秒内推送到目标客户端。
   1. 质量属性
      1. **安全性**
9. 需要保障用户账户安全。
10. 需要保障服务器的数据安全,包括数据库和用户上传文件；
11. 需要保障通信安全，通信数据需要加密，防止传递的信息被他人监听并轻松得到信息的内容；
    * 1. **可靠性**
12. 软件在用户输入极端数据或恶意操作时候，有充分的合法性检查，可以避免崩溃或至少不会殃及其他用户。
13. 程序报错时有效提醒用户错误产生的可能原因以及可以采取的措施，生成错误报告，上传到服务器，以便对报错进行重现、分析、纠错。
    * 1. **易用性**
14. 图标醒目易懂，尽量辅助以文字说明。
15. 流程尽量简化。
    * 1. **可维护性**
16. 代码命名规范。
17. 代码注释详略得当。
18. 对代码进行重构。
    1. 对外接口
    2. 约束

# 概要设计

* 1. 网络结构设计

软件采用C/S结构，其中客户端部分为安卓智能手机，通过互联网和服务器进行通信，服务器主要涉及到四个：Web服务器、Socket服务器、QQ邮件服务器、极光推送服务器，如图3.1。

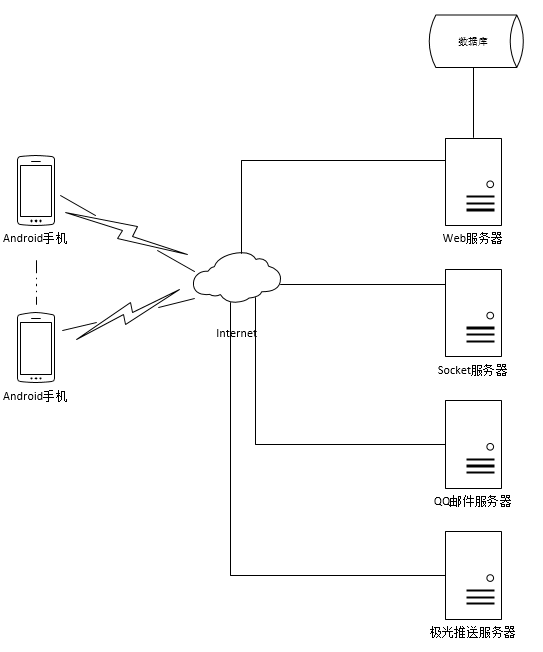


图3.1 系统网络拓扑图

四个服务器的作用如下：

1. Web服务器：处理客户端发送的HTTP请求，包括授权访问检查、业务逻辑处理、文件上传下载、数据库操作等。
2. Socket服务器：用户进行白板绘画、会内群聊的时候，每个客户端都和Socket服务器之间维持着一个WebSocket连接，用来进行数据和消息的实时同步。Socket服务器就是负责处理Socket请求，单发、群发、广播消息的。
3. QQ邮件服务器：用来给用户发送邮箱验证码。
4. 极光推送服务器：用于推送联系人添加、接受、拒绝、删除、邀请加会请求到客户端。
   1. 功能结构设计

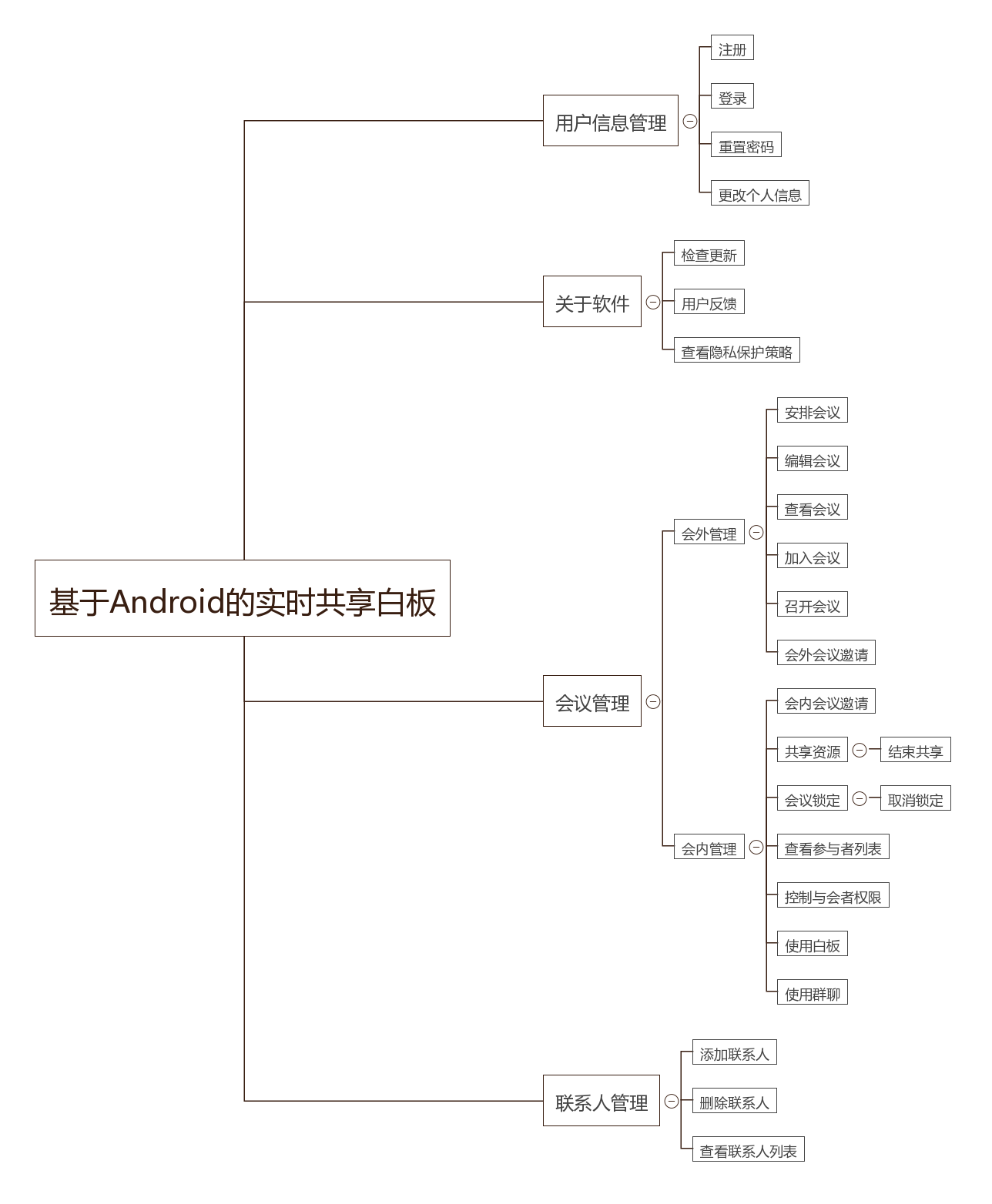


图3.2 系统功能结构图

* 1. 数据库设计

服务器端数据库采用MySql，客户端数据库采用SQLite。

MySql 中数据库包括5张表，见表3.3.1到表3.3.5。

表3.3.1 用户表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 长度 | 索引 | 注释 |
| user\_id | INT | 10 | 主键 | 用户ID |
| user\_email | VARCHAR | 30 | 主键 | 邮箱 |
| user\_family\_name | VARCHAR | 15 |  | 姓 |
| user\_given\_name | VARCHAR | 15 |  | 名 |
| user\_register\_time | DATETIME | 32 |  | 注册时间 |
| user\_login\_recent\_time | DATETIME | 32 |  | 最近一次登陆时间 |
| user\_password | VARCHAR | 32 |  | 登录密码 |
| user\_token | VARCHAR | 32 |  | Token |
| user\_avatar | VARCHAR | 60 |  | 用户头像地址 |

表3.3.2 会议信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 长度 | 索引 | 注释 |
| meeting\_id | INT | 10 | 主键 | 会议ID |
| meeting\_url | BIGINT | 20 | 主键 | 会议号 |
| meeting\_theme | VARCHAR | 20 |  | 会议主题 |
| meeting\_host\_user\_id | INT | 10 | 外键 | 主持人用户ID |
| meeting\_is\_drawable | TINYINT | 4 |  | 与会者默认能否使用白板绘画： 1：不能 2： 可以 |
| meeting\_is\_talkable | TINYINT | 4 |  | 与会者默认能否聊天： 1：不能 2： 可以 |
| meeting\_is\_add\_to\_calendar | TINYINT | 4 |  | 是否添加到日历提醒：1：不添加 2：添加 |
| meeting\_password | VARCHAR | 32 |  | 登录密码 |
| meeting\_start\_time | DATETIME | 32 |  | 会议预期开始时间 |
| meeting\_end\_time | DATETIME | 32 |  | 会议预期结束时间 |
| meeting\_status | TINYINT | 4 |  | 会议状态：  1：未开始并且未到期 2：未开始并且过期了 3：正在进行 4：开会结束  5：锁定 |
| event\_id | BIGINT | 20 |  | 日历事件ID |
| meeting\_desc | TEXT |  |  | 会议描述 |

表3.3.3 用户参加会议表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 长度 | 索引 | 注释 |
| user\_and\_meeting\_id | INT | 10 | 主键 | 用户参加会议ID |
| check\_in\_type | TINYINT | 4 |  | 用户参会类型：1 ：与会 2：主持会议 |
| check\_in\_time | DATETIME | 32 |  | 最新入会时间 |
| check\_out\_time | DATETIME | 32 |  | 最新离会时间 |
| bd\_user\_user\_id | INT | 10 | 外键 | 参会者的用户ID |
| bd\_meeting\_meeting\_id | INT | 10 | 外键 | 参会者进入会议的会议ID |

表3.3.4 联系人信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 长度 | 索引 | 注释 |
| friend\_id | INT | 10 | 主键 | 联系人ID |
| response\_status | TINYINT | 4 |  | 回复状态：1：未回复 2：拒绝 3：同意 4.好友关系已经删除 |
| message\_time | DATETIME | 32 |  | 消息时间 |
| bd\_user\_user\_id | INT | 10 |  | 用户1的ID |
| bd\_user\_user\_id1 | INT | 10 |  | 用户2的ID |

表3.3.5 版本更新表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 长度 | 索引 | 注释 |
| version\_id | INT | 10 | 主键 | 用户ID |
| app\_name | VARCHAR | 32 |  | 应用名称 |
| server\_version | VARCHAR | 32 |  | 服务器端APP版本号 |
| last\_force | TINYINT | 4 |  | 是否强制更新：0：不强制 1：强制 |
| server\_flag | TINYINT | 4 |  | 标记 |
| update\_url | TEXT |  |  | 新版本APP下载地址 |
| upgrade\_info | TEXT |  |  | 更新提示信息 |
| update\_time | DATETIME | 32 |  | 版本发布时间 |

SQLite中数据库包括一张表：如表3.3.6 。

表3.3.6 系统消息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 长度 | 索引 | 注释 |
| \_id | INTEGER | 根据值的大小存储在 1、2、3、4、6 或 8 字节中 | 主键 | 联系人ID |
| EMAIL | TEXT |  |  | 用户邮箱 |
| TITLE | TEXT |  |  | 消息标题 |
| CONTENT | TEXT |  |  | 消息内容 |
| FAMILY\_NAME | TEXT |  |  | 用户的姓 |
| GIVEN\_NAME | TEXT |  |  | 用户的名 |
| FEATURE | TEXT |  |  | 消息类别 |
| AVATAR | TEXT |  |  | 用户头像地址 |
| STATUS | INTEGER | 根据值的大小存储在 1、2、3、4、6 或 8 字节中 |  | 消息状态：0：初始态 1：已同意 2：已拒绝 |
| MSG\_TIME | INTEGER | 根据值的大小存储在 1、2、3、4、6 或 8 字节中 |  | 消息到达时间 |
| TAG | TEXT |  |  | 消息全局唯一标识：用户邮箱+时间戳毫秒值 |

* 1. 综合考虑
     1. **稳定性和可扩展性**
     2. **性能分析**
     3. **安全性**
     4. **其它**

# 详细设计

* 1. 模块设计
  2. 接口设计
     1. **UI设计**

1. 系统至少可以维持300个WebSocket并发连接。
2. 系统至少可以维持20个容量为20人的会议。
   * 1. **内部接口设计**
3. 系统至少可以维持300个WebSocket并发连接。
4. 系统至少可以维持20个容量为20人的会议。
   * 1. **外部接口设计**
5. 系统至少可以维持300个WebSocket并发连接。
6. 系统至少可以维持20个容量为20人的会议。

结论

结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论结论。

cjrg创建日期

ddzt订单状态

ddjg订单价格

ffcrq发车日期

fpzs票张数

fpmm取票密码

fplx票类型

fqph取票号

ckxm乘客姓名

sfzhm身份证号码

lxdh联系电话

mdddm到达站代码

致谢

致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢致谢

【参考文献】

1. 参考文献1
2. 参考文献2
3. 参考文献3
4. 参考文献4 ……

附录：