南京邮电大学通达学院毕业设计(论文)开题报告

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 题 目 | | 基于Android的聊天系统的设计与实现 | | | | |
| 学生姓名 | | 项伟伟 | 班级学号 | 18240125 | 专业 | 软件工程(嵌入式培养) |
| 1. 课题任务的学习与理解 2. 课程背景调查   自即时通讯(IM)软件诞生以来, 其便利性受到社会各界的青睐。典型的代表为微信、QQ等。即时通讯比传统电子邮件所需时间更短，且较电话更方便。其主要特点为: 多任务作业、异步、长短沟通、媒介转换迅速、高交互性、不受时空限制。  早期的即时通讯软件只能进行文本、预设的图片、文件的交流, 依靠服务器进行缓存。如QQ, 其早期会员功能可以支持服务器长期缓存聊天数据, 而后诞生的微信, 则只采用短时缓存的方式, 不持久化保存聊天数据。伴随移动互联网的发展和Cov19的时代背景, 即时通讯服务开始提供会议、VoIP。各种媒介的边界因为即时通讯而变得模糊。  另一方面, 由于当今各大互联网企业的相关业务的发展, 即时通讯软件已成为集生活服务、社交、娱乐等于一身的功能性软件, 其冗余的功能备受争议, 如QQ移动端集成了虚幻SDK(Unreal SDK)。本设计将实现一个精简、小巧而纯粹的即时通讯软件。   1. 阅读文献资料进行调研的综述   **C/S架构**  绝大多数即时通讯软件采用了C/S架构, 即客户端/服务器体系结构。在即时通讯系统中, 通常客户端上的某人连上IM服务器时发出信息通知另一个客户端的某个使用者，双方可透过互联网开始进行实时的通讯。客户端和服务器端的程序不同，用户的程序主要在客户端，服务器端主要提供数据管理、数据共享、数据及系统维护和并发控制等，客户端程序主要完成用户的具体的业务。C/S主要特点是交互性强、具有安全的存取、响应速度快、利于处理大量数据。但是C/S结构相比B/S缺少通用性，系统维护、升级需要重新设计和开发，增加了维护和管理的难度，进一步的数据拓展困难较多。  **TCP协议**  传输控制协议（TCP，Transmission Control Protocol）是一种面向连接的、可靠的、基于字节流的传输层通信协议。本设计主要利用TCP协议的可靠性。  TCP在传输数据时，会将数据拆分为适当大小的报文段，其中报文首部需占20字节。报文首部包括了源端口号、目的端口号、序号、确认号、数据偏移、标志位、窗口、校验和、紧急指针和选项。能够在传输层唯一确定用户使用的APP。其次，TCP主要利用如下手段确保报文的可靠性：   1. 数据分块: 应用数据被分割成TCP认为合适的报文大小。 2. 序列号和确认应答：TCP会给发送的每一个包进行编号，在传输的过程中，每次接收方收到数据后，都会对传输方进行确认应答（即回复ACK报文），这个 ACK 报文当中带有对应的确认序列号（即回复序号 = 接收序号 + 1），告诉发送方成功接收了哪些数据以及下一次的数据从哪里开始发。除此之外，接收方可以根据序列号对数据包进行排序，把有序数据传送给应用层，并丢弃重复的数据。由于TCP能提供全双工通信，因此通讯双方都不必专门发送确认报文，只需要在传输数据时将确认信息捎带，大大提高了传输效率。 3. 校验和：TCP 将保持它首部和数据部分的检验和。这是一个端到端的检验和，目的是检测数据在传输过程中的任何变化。如果收到报文段的检验和有差错，TCP 将丢弃这个报文段并且不确认收到此报文段，服务端在一段时间内没收到确认报文段，将转入超时重传机制。 4. 超时重传机制：超时重传机制最关键的因素是重传计时器的设定。由于互联网是非常复杂的环境，可能某一个时段的媒介是高速局域网，下一个时段的媒介是低速的广域网等。TCP采用了一种自适应的算法。其思想描述如下：记录每一个报文发出的时间，以及收到相应的确认报文的时间，这两个时间的差就是报文的往返时延。首先将采样报文的往返时延样本，将各个报文的往返时延加权平均，得到报文段的加权平均往返延迟RTT，每测量到一个新的往返延时样本就取一次平均。新RTT=α×（旧的SRTT）+（1-α）×（新的RTT样本）， 0≤α<1且典型的α为7/8。此外还有RTT的偏差的加权平均值。新RTTD=β×（旧的RTTD）+（1-β）× |新RTT - 新的RTT样本|。0≤β<1且典型的β为3/4。最后，超时重传时间RTO = 新RTT + 4 × 新RTTD 5. 流量控制：TCP 通过滑动窗口协议来实现流量控制机制。连接的双方都有一个固定大小的缓冲空间，发送窗口在连接建立时由双方商定初始值。在通信的过程中，接收端可根据自己的资源，随时动态调整接收窗口，然后通知发送方，使得收发双方的窗口一致。防止产生丢包。 6. 拥塞控制：当网络节点出现拥塞时，减少数据的发送。TCP为了进行有效的拥塞控制通过拥塞窗口来进行控制。发送方的原则是：只要网络没有出现拥塞，拥塞窗口就可以再增大一点(一般是二倍扩大)，其拥有四个手段 7. 慢启动：TCP连接建立或网络发生拥塞超时，将拥塞窗口设置为一个报文大小 8. 拥塞避免：当拥塞窗口的大小≥慢启动阀值，进入拥塞避免，线性增加拥塞窗口的大小。 9. 快速重传：发送方如果连续收到对同一报文三个重复确认的ACK，则立即重传该报文段，不等到重传计时器的超时。触发快速重传后，慢启动阈值设置为原先的一半。 10. 快速恢复：当采用快速重传时直接执行拥塞避免算法。 11. ARQ协议：原则是发送方发送的数据量不能超过接收端缓冲区的大小。当接收方来不及处理发送方的数据，会提示发送方降低发送的速率，防止产生丢包。ARQ协议又分为连续ARQ协议和选择ARQ协议。前者的发送方维持着一个一定大小的发送窗口，位于发送窗口内的所有分组都可连续发送出去，而中途不需要等待对方的确认。而发送方每收到一个确认就把发送窗口向前滑动一个分组的位置。这样信道的利用率就提高了。当接收方无法接受到内容后，需要再退回已经确认收到的后一个分组进行重传。后者可以用作一个消息单元传送和确认的协议。当用作传送消息单元的协议时，发送进程根据一个指定大小的 窗口持续发送若干帧 ，即使发送过程中丢失帧，也会继续发送。 12. 初步拟定的执行方案（含具体进度计划） 13. 初步执行方案   C端方案  方案的C端采用TCP+安卓+Kotlin开发。整体分为六个层，自顶向下分别是：前端UI层、展示层、业务层、运行时层、数据源层、底层运行环境。  前端UI层：负责向用户展示，承担了绝大多数的UI。  业务层：负责内容的处理和数据请求操作。当用户在前端发出指令，业务层负责分析执行相关内容，如果缺少数据，将向下层请求数据。最后向UI层发送执行结果，UI层向用户展示反馈。  运行时层：运行时层承担了绝大多数重要的工作。运行时层(数据)向业务层提供数据内容的API，向下管理各个本地数据，向远程服务器端发送接收信息。运行时层(渲染)负责渲染完整的界面、提供界面模板等。运行时层(异常、日志)处理整个客户端的异常和用户日志。  数据源层：分为本地数据源和网络数据源。本地数据源一般是网络数据源的缓存，为了加快速度，提高用户体验。网络数据源被获取后会在本地进行存储，用户可以选择性删除。  运行环境层：模糊不同运行环境的差异。    B端方案  方案的B端采用TCP + Java/Go/C++ 开发。整体分为服务端收发端、业务层、管理层、数据源层、日志管理、异常管理。     1. 具体计划进度 2. 学习了解移动互联网相关理论知识, 提出总体设计方案, 分析系统网络架构, 完成开题报告   2周   1. 熟练掌握和使用Android开发环境, 尽快掌握Android工作流程   2周   1. 搜索并学习类似软件系统的技术架构和开发方法   3周   1. 掌握java语言互联网通信技术的底层原理及相关的具体代码实现   3周   1. 深入了解安卓系统特性, 优化UI界面, 优化软件使用过程及细节, 增强与系统中其他APP的互动性   4周   1. 进一步完善系统功能, 并系统进行整体测试   1周   1. 整理资料, 论文写作, 准备答辩   1周   1. 附录   主要参考文献和资料  [1]皮成.基于Android平台的即时通信中间件的研究与实现[D].西安电子科技大学,2014.1-62.  [2]袁远.基于Android平台端到端即时通信系统的分析与设计[D].北京邮电大学,2012.1-67.  [3]吴亚峰. Android应用案例开发大全[第三版]. 北京. 人民邮电出版社, 2015.  [4]郭霖.第一行代码 Android [第三版].北京. 人民邮电出版社,2020.  [5]佘志龙, 陈昱勋, 郑名杰, 陈小凤．Google Android SDK开发范例大全3[M]．北京：人民邮电出版社, 2011．  [6]纳德尔曼．Android应用UI设计模式[M]．袁国忠, 译．北京：人民邮电出版社, 2013.  [7]丰生强．Android软件安全与逆向分析[M]．北京：人民邮电出版社,2013.  [8]Qi Y, Cao M, Zhang C, et al. A Design of Network Behavior-Based Malware Detection System for Android[M]. Algorithms and Architectures for Parallel Processing. Springer International Publishing, 2014.  [9]Arzt S, Rasthofer S, Fritz C, et al. FlowDroid: Precise Context, Flow, Field, Object-sensitive and Lifecycle-aware Taint Analysis for Android Apps[J]. Acm Sigplan Notices, 2014, 49(6), 259-269. | | | | | | |
| 指导教师批阅意见 | 指导教师(签名)： 2022 年 3 月 日 | | | | | |