Atitit 客服线im计划技术规划

目录

[1. 基本目标 客服线 3](#_Toc29257)

[1.1. 基本的im通讯与 问答系统 业务知识库系统 3](#_Toc6659)

[1.2. 普通版本和lite版 4](#_Toc28199)

[1.3. 预计使用人员数量 10w+ 4](#_Toc15095)

[1.4. 服务端跨平台部署（ 4](#_Toc14143)

[1.5. 简单易得 快速部署 4](#_Toc7916)

[2. 大略功能 4](#_Toc1097)

[2.1. 显示客服列表 4](#_Toc10548)

[2.2. 文字类消息 4](#_Toc31965)

[2.3. 消息转发 回复 5](#_Toc13905)

[2.4. 消息翻译 5](#_Toc14746)

[2.5. 离线消息 留言消息功能 5](#_Toc31232)

[2.6. 大段消息摘要 5](#_Toc3704)

[2.7. 消息检索 快速检索+高级检索 5](#_Toc29381)

[2.8. 繁简转换 与拼音转换 5](#_Toc14510)

[2.9. 消息的紧急程度归类标识 优先级提升 5](#_Toc18098)

[2.10. 消息词云 5](#_Toc5339)

[2.11. 对话管理切分 ，按照时间段自动切分对话，方便查询 6](#_Toc15224)

[2.12. 对话问题处理 6](#_Toc26441)

[2.13. 消息手动归类 主题化 与tag标记 6](#_Toc19468)

[2.14. 消息的自动归类 6](#_Toc20363)

[2.15. 相似问题 拓展 6](#_Toc12389)

[2.16. 问题与答复归类 6](#_Toc2500)

[2.17. 多媒体 图像 和 音频的检索 6](#_Toc4115)

[2.18. 文件传输与传图 7](#_Toc29520)

[2.19. 表情包系列 7](#_Toc7884)

[2.20. 语音系列语音 7](#_Toc13746)

[2.21. 图像简单涂鸦修改编辑 7](#_Toc6686)

[2.22. 历史纪录导出 7](#_Toc20976)

[2.23. 文件的与多媒体文件的管理与归类 7](#_Toc6627)

[2.24. 导出 导入 备份 8](#_Toc22467)

[2.25. 消息撤回 删除 8](#_Toc28974)

[2.26. 消息阅后即焚功能，支持数据自定义加解密存储。 8](#_Toc12094)

[2.27. 地图LBS相关功能 8](#_Toc3017)

[2.28. 群组功能 8](#_Toc20712)

[2.29. 视频系列（方便指导用户某些操作 8](#_Toc21629)

[2.30. 人脸消除功能 安全系列，自动将人脸马赛克或黑洞轮廓化 8](#_Toc26451)

[2.31. 可以在拍照和视频方面应用起来 8](#_Toc21179)

[2.32. 安全音视频 变声 消除声纹 与人脸 8](#_Toc28343)

[2.33. 图像提取文字提取ocr 8](#_Toc25178)

[2.34. 消息 tag标注 与分类 8](#_Toc18972)

[2.35. 二维码识别添加人员 ，跳转url等 8](#_Toc16010)

[2.36. 文件预览 压缩文件 pdf 等 8](#_Toc17170)

[2.37. 文本提取与检索 从各种传输文件里面 8](#_Toc8499)

[2.38. 消息文字校对 错别字词更正提示 10](#_Toc20501)

[2.39. 二阶段特性 10](#_Toc588)

[3. 后端管理功能表 10](#_Toc7939)

[3.1. 用户管理 10](#_Toc7924)

[3.2. 消息主题库 与主题的大分类汇集 10](#_Toc3155)

[3.3. 消息管理 10](#_Toc28242)

[3.4. 普通文件管理 10](#_Toc13547)

[3.5. 回话管理 10](#_Toc6800)

[3.6. 多媒体类文件管理 10](#_Toc32712)

[3.7. 消息主题与分析 10](#_Toc1627)

[3.8. 一些统计分析规划 10](#_Toc9692)

[3.9. 其他 10](#_Toc27612)

[4. 标准消息格式规范与传输模式 10](#_Toc12739)

[4.1. STOMP 协议面向简单文本的消息传递协议 11](#_Toc18439)

[4.2. webSocket进阶篇——STOMP Over Websocket 11](#_Toc32323)

[4.3. 消息的传输可以走websocket 12](#_Toc7588)

[5. 技术规化 12](#_Toc12726)

[5.1. 常见开发语言 12](#_Toc13711)

[5.2. Ui可使用h5 ，以及某些私有化xml，混合应用也可适当使用原生控件 13](#_Toc26058)

[5.3. 交互 websocket reset ajax 13](#_Toc28153)

[5.4. 数据存储 13](#_Toc26172)

[5.5. 开发框架与模块api rest框架/socket框架 13](#_Toc30898)

[5.5.1. websocket模块 14](#_Toc1660)

[5.5.2. Librosa 音频处理 14](#_Toc10440)

[5.5.3. Opencv 视频与图像处理 14](#_Toc19802)

[5.5.4. Ffmpeg 音视频处理 14](#_Toc12787)

[5.5.5. H5的图像 音频系列api 14](#_Toc28458)

[5.5.6. 各个平台提供的图像 音视频api 14](#_Toc4273)

[5.6. 该选择什么样的网络通讯技术？ 14](#_Toc4613)

[5.7. OCR）引擎。Tesserac 15](#_Toc18433)

[5.8. Nlp snowNLp 15](#_Toc29966)

[5.9. Es全文检索 15](#_Toc18965)

[5.9.1. 电子名片协议vCard 处理 15](#_Toc6885)

[5.10. 微格式与 （geo地理标识 15](#_Toc19319)

[6. --------------需求之FURPS模型 15](#_Toc22703)

[7. 技术类功能核心 16](#_Toc31857)

[7.1. 消息格式规范可以使用STOMP 的即时通讯消息规范 16](#_Toc30297)

[7.2. 消息传输事件驱动websocket 16](#_Toc17093)

[7.3. 定时可用 db定时以及os定时机制，加强稳定性 16](#_Toc12804)

[7.4. 文字分析与全文搜索类等Nlp功能 16](#_Toc7659)

[7.5. 其他 16](#_Toc18363)

[7.5.1. 小文件可以使用base64走消息通道，，大文件可走h5文件传输或其他通道 16](#_Toc29294)

[7.5.2. Html5已经支持http断点续传上传 17](#_Toc26588)

[8. 可扩展性 结构简化可维护性 17](#_Toc7467)

[8.1. 简单设计 17](#_Toc6629)

[8.2. 业务逻辑设计简化复杂分支 17](#_Toc30069)

[8.3. 动态数据结构 与扩展性 17](#_Toc13950)

[8.4. 适当的层次结构 17](#_Toc23916)

[8.5. 增加sql支持 17](#_Toc15092)

[8.6. 存储云服务化简化开放与运维 17](#_Toc14865)

[9. 稳定性 17](#_Toc2444)

[9.1. 进程保活 cron保活 17](#_Toc14547)

[9.2. 分布式 非集中模式 分平台分片部署 18](#_Toc1356)

[9.3. 基于现有db和os机制更佳可靠性 18](#_Toc13992)

[9.4. 资源自动释放gc vs 永驻内存 18](#_Toc11565)

[9.5. 全面边界检测与异常管理 18](#_Toc31999)

[9.6. 对数据查询限制最大单次条数 18](#_Toc14812)

[9.7. 心跳保活 18](#_Toc27387)

[9.8. 文件中转模式 18](#_Toc6762)

[9.9. 服务器中转方式 19](#_Toc5601)

[10. 未来性能方面 19](#_Toc20387)

[10.1. 负载均衡 nginx （dubbo》》springcloud） 19](#_Toc415)

[10.2. 分片区存储 分区 分库分表 19](#_Toc25305)

[11. 安全性 19](#_Toc23246)

[12. 附属的高级功能 20](#_Toc23666)

[12.1. Sql的支持 20](#_Toc9644)

[13. 附注： 20](#_Toc32633)

[13.1. 表情包表达分类体系树 20](#_Toc12754)

[14. Ref参考 20](#_Toc31476)

# 基本目标 客服线

## 基本的im通讯与 问答系统 业务知识库系统

后期增加智能聊天系列

通讯的三种目的类型：QA问答型、任务型 和 知识获取型，其他

## 普通版本和lite版

Lite 版本实现有web pwa hybrid 和快应用，各大平台小程序等

GUI客户端 web +pwa+ 跨平台移动hybrid app

## 预计使用人员数量 10w+

满足中小型组织使用

人员较少 暂无需特别性能提升

## 服务端跨平台部署（

服务的部署在Linux Win，。

客户端方面基本lite版本都是跨平台的，本地版本可以支持android ios平台等

## 简单易得 快速部署

满足中小型组织使用。。容器化部署。。。 上传解压运行绿色版，无需太繁琐安装与配置

# 大略功能

## 显示客服列表

## 文字类消息

发送消息 接受消息

在线消息与离线消息

## 消息转发 回复

## 消息翻译

## 离线消息 留言消息功能

## 大段消息摘要

消息很长的时候，将其摘要

## 消息检索 快速检索+高级检索

消息查询简单查询 （关键词查询法，日期查询 法，

消息检索搜索 高级查询（消息类型【发送/接受】，消息接收人，日期，日期区段，消息内容 返回条数等）

文字NLP近义词扩展搜索（常常会遇到这种情况，我们大概知道消息有哪些内容意思，但是不能想起具体的关键词导致不能搜索到，这时候可以使用意思相近的词来扩展搜索。。

关键词提取

## 繁简转换 与拼音转换

## 消息的紧急程度归类标识 优先级提升

某些消息需要优先处理。。 自动化归类与手动归类，原则上应该90%可以自动化归类，手动在少量处理。。

## 消息词云

自动关键词提取，归类，可以按照归类展示词云

或直接按照词语展示云 ，按照词频率排列，可以很容易得出用户关注的问题排行榜

## 对话管理切分 ，按照时间段自动切分对话，方便查询

比如讲某个时间段的自动归类为一个对话。。。

也可以手动跨长时间段调整对话时长切分 ，更加精确

比如今天谈论了几轮谈话，就是几个对话。。。

## 对话问题处理

讲多条消息总结为一个问题对话，可将其状态设置为处理中 ，完毕等。。

## 消息手动归类 主题化 与tag标记

支持范围归类，对某轮对话 将其归类到某个主题

## 消息的自动归类

根据消息的主题归类 谈论的事务 。。。

## 相似问题 拓展

## 问题与答复归类

将具有同一类似问题的答复收集，成为回复列表。。可以评选最佳答案的消息

## 多媒体 图像 和 音频的检索

可以自动提取其中的tag标注元数据。。

可以，体积大小，时间段搜索，发送人/接收人

图像的搜索，可以根据分辨率大小 ，颜色（七大色）主体搜索，类型（网络图片，照片，gif等），gif时长，，清晰度

可以根据时长，

## 文件传输与传图

## 表情包系列

具备了语音、文本、表情、传图等基础IM业务功能

表情包 sticker贴纸 gif emoji Im系统 表情包系列体系树与分类

## 语音系列语音

截屏 文件传输

语音留言+播放功能 语音转文字 文字转语音TTS 变声 声纹叠加

音视频支持 **实时音频 语音留言**

**背景去噪** 循环功能 语音识别（asr） 波形展示预览

静音掠过（直接跳过无音语音片段

## 图像简单涂鸦修改编辑

图像截屏与简单修改功能（文字标注，圈阅，箭头涂鸦 等）

Emoji图标

Ffmpeg语音编码

## 历史纪录导出

## 文件的与多媒体文件的管理与归类

## 导出 导入 备份

## 消息撤回 删除

## 消息阅后即焚功能，支持数据自定义加解密存储。

## 地图LBS相关功能

## 群组功能

## 视频系列（方便指导用户某些操作

## 人脸消除功能 安全系列，自动将人脸马赛克或黑洞轮廓化

## 可以在拍照和视频方面应用起来

## 安全音视频 变声 消除声纹 与人脸

## 图像提取文字提取ocr

## 消息 tag标注 与分类

## 二维码识别添加人员 ，跳转url等

## 文件预览 压缩文件 pdf 等

## 文本提取与检索 从各种传输文件里面

3.1. Ocr 图片提取文字 2

3.2. Tts语音提取文字 语音识别（Speech recognition） 2

3.3. Eml处理 2

3.4. 压缩文件处理 zip rar 2

3.5. Office系列文档提取 2

3.6. Pdf提取信息 2

文本提取

## 消息文字校对 错别字词更正提示

## 二阶段特性

## 结构化数据

比如发送url消息 ，可以展示为标题，摘要模式

Vcard名片 ，地图类的 ，活动日程类的

# 后端管理功能表

## 用户管理

## 消息主题库 与主题的大分类汇集

## 消息管理

## 普通文件管理

## 回话管理

## 多媒体类文件管理

## 消息主题与分析

## 一些统计分析规划

## 其他

# 标准消息格式规范与传输模式

文本协议比较好 方便调试 ，支持web通道http

自定义规范消息格式

## STOMP 协议面向简单文本的消息传递协议

STOMP 的js java支持也很好，很多资料和类库

## webSocket进阶篇——STOMP Over Websocket



背景介绍

之前提到使用原始的[websocket](https://www.jianshu.com/p/326290d38abe" \t "https://www.jianshu.com/p/_blank)，实现后台消息的主动推送，但是这种方式过于偏向底层，需要开发人员去手动的保存用户连接到websocket中的信息，这个信息不仅仅是用户的id、name而已还要保存他们的订阅信息，因为完完全全有可能所有已连接用户需要推送的消息是不一样的，而且可能一个用户会订阅很多的推送信息，比如说：在一个新闻网页中，有的用户对军事感兴趣，有的用户对科技感兴趣，有的对开源的代码感兴趣，而有的可能对所有的都感兴趣，如果些东西需要根据数据库实时更新的话，在使用原始的websocket来管理就会变成十分麻烦。

基于STOMP协议的WebSocket

使用STOMP的好处在于，它完全就是一种消息队列模式，你可以使用生产者与消费者的思想来认识它，发送消息的是生产者，接收消息的是消费者。而消费者可以通过订阅不同的destination，来获得不同的推送消息，不需要开发人员去管理这些订阅与推送目的地之前的关系，spring官网就有一个简单的spring-boot的stomp-demo,如果是基于springboot，大家可以根据spring上面的教程试着去写一个简单的demo。

作者：rpf\_siwash

****rest使用HTTP轮循方式****

简介：定时向HTTP服务端接口（Web Service API）获取最新消息。  
优点：实现简单、可控性强，部署硬件成本低。  
缺点：实时性差

## 消息的传输可以走websocket

# 技术规化

## 常见开发语言

开发语言方面主体使用js 会更加简化与开发效率提升。。可以辅以java python调用少量某些特殊类库api

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 开发效率 | 简单性 | 跨平台 | 需要编译 | 需要部署 | 已知问题 |
| Sql | 高 | 简单 | Y | N | N | 非全能DSL  调试支持不足，for循环语法啰嗦，异常处理啰嗦 |
| php | 中 | 中等 | Y | N | Y | 调试支持没有内建 |
| java | 低 | 繁琐 | Y | Y | Y | 语法啰嗦繁琐 |
| Node.js | 中 | 中等 | Y | N | Y | 单线程影响稳定性 |
| go语言 | ？？ | ？？ | N | Y  区分win linux平台 | Y | 异常处理繁琐，影响稳定性.需要区分平台编译 |
| python | 中 | 中等 | y | N | y | Web使用不多 |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

## Ui可使用h5 ，以及某些私有化xml，混合应用也可适当使用原生控件

## 交互 websocket reset ajax

业务普通走rest通道，消息及时性要求高的走websocket通道。。非web情况下也可以走其他通道，扩展性强

## 数据存储

服务的数据存储Mongodb存储更加方便且性能优越。。Es可存储全文检索消息数据

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Sql支持dsl支持 | 复杂业务支持 |
| mysql |  | 支持 | 高 |
| mongodb |  | Nosql 但有dsl | 中 |
| es |  | Dsl功能较薄弱 | 低 |
| redis |  | 啥都没 | 最低 |
|  |  |  |  |
|  |  |  |  |

客户端数据存储普通使用h5的store即可。有高级需求可使用sqlite等

## 开发框架与模块api rest框架/socket框架

前端直接支持websocket api，，后端使用各自语言的websocket模块类库

Rest api前端可使用featch api，后端使用各自语言的rest模块或类库

### websocket模块

### Librosa 音频处理

### Opencv 视频与图像处理

### Ffmpeg 音视频处理

### H5的图像 音频系列api

### 各个平台提供的图像 音视频api

## 该选择什么样的网络通讯技术？

****IM主流网络通讯技术有两种:****

* 基于TCP的长连接；
* 基于HTTP短连接PULL的方式。

基于TCP长连接则能够更好地支持大批量用户，问题是客户端和服务器的实现比较复杂。当然也还有一些变种，如下行使用MQTT进行服务器通知/消息的下发，上行使用HTTP短连接进行指令和消息的上传。这种方式能够保证下行消息/指令的及时性，但是在弱网络下上行慢的问题还是比较严重。早期的来往就是基于这种方式。

短连接PULL常见于WEB IM系统(当然现在很多WEB IM都是基于WebSocket实现)，它的优点是实现简单，方便开发上手， ，比较适合小型的IM系统

## **OCR**）引擎。Tesserac

## Nlp snowNLp

## Es全文检索

mongo3.4 版本支持中文检索，所以直接使用mongo的检索，效率很高。可以通过db.create\_index([("metaDataList.title",pymongo.TEXT)]) 建text索引，metaDataList.title是字段名称，也可以同时对好几个字段做索引，用db.create\_indexes()，索引建好后  
db.find({ "$text": { "$search": "张三的" }},{ "score": { "$meta": "textScore" } })  
是通过你输入的关键词，返回的文档的得分textScore，

你也可以用  
db.aggregate([  
{ "$match": { "$text": { "$search": "张三" } } },

{ "$sort": { "score": { "$meta": "textScore" } } },

{ "$project":

{ "metaDataList.skuid": 1, "metaDataList.product\_info": 1,

"metaDataList.title": 1, "metaDataList.url": 1,

"\_id": 1 } },

{"$limit": 50}

])查询

### **电子名片协议**vCard 处理

vCard是一种现存的、广泛使用的，用户个人信息存储的标准，有点像是电子名片。基础的功能是存储和获取用户的电子身份，该信息是用XML表示的，数据的存储取决于所有现存的Jabber服务器的实现。

导入导出

## 微格式与 （[geo](https://zh.wikipedia.org/wiki/Geo" \o "Geo)地理标识

# --------------需求之FURPS模型

**FURPS**是功能（function）、易用性（usability）、可靠度（reliability）、性能（performance）及可支援性（supportability）五个词英文字首的缩写，是一种识别软体品质属性的模型。

# 技术类功能核心

## 消息格式规范可以使用STOMP 的即时通讯消息规范

## 消息传输事件驱动websocket

## 定时可用 db定时以及os定时机制，加强稳定性

## 文字分析与全文搜索类等Nlp功能

## 其他

音频语音类功能

Ffmpeg视线音频类的编辑 按照时间点切割

图像类的修改与编辑

图像的合成，双图像叠加 ，文字图像合成叠加

图像涂鸦 ，截屏 ，编码格式转换

抠图(手动涂鸦式抠图） 马赛克

技术实现

一个最强大的图像处理库ImageMagick，它被誉为命令行上的PhotoShop，而且被各大公司 ...

二进制文件的传输 [Base64](https://zh.wikipedia.org/wiki/Base64" \o "Base64)编码 vs 单独二进制通道

### 小文件可以使用base64走消息通道，，大文件可走h5文件传输或其他通道

**二进制资料传输受限**

XMPP传输单一的XML文件，因此要透过XMPP传输二进制资料，需先将二进制资料以[Base64](https://zh.wikipedia.org/wiki/Base64" \o "Base64)编码。但是大量的二进制资料传输，较佳的作法是用外部的其他协议传输，而以XMPP进行沟通。至于其他二进制资料加密会话（encrypted conversations）或图形图标（graphic icons）以嵌入式使用相同的方法。

### Html5已经支持http断点续传上传

html5的File对象的 slice 可以用于从文件切取片段来上传。

# 可扩展性 结构简化可维护性

## 简单设计

## 业务逻辑设计简化复杂分支

## 动态数据结构 与扩展性

## 适当的层次结构

## 增加sql支持

## 存储云服务化简化开放与运维

# 稳定性

## 进程保活 cron保活

**保活处理**

## 分布式 非集中模式 分平台分片部署

## 基于现有db和os机制更佳可靠性

## 资源自动释放gc vs 永驻内存

## 全面边界检测与异常管理

## 对数据查询限制最大单次条数

防止无响应。。熔断拒绝机制

## 心跳保活

心跳保活机制的实现方案参考

从上面我们可以得出结论，目前而言，应用层心跳的确是检测连接有效性，双方是否存活的最佳实践，那么剩下的问题就是怎么实现。  
  
最简单粗暴做法当然是定时心跳，如每隔 30 秒心跳一次，15 秒内没有收到心跳回包则认为当前连接已失效，断开连接并进行重连。这种做法最直接，实现也简单。唯一的问题是比较耗电和耗流量。以一个协议包 5 个字节计算，一天收发 2880 个心跳包，一个月就是 5 \* 2 \* 2880 \* 30 = 0.8 M 的流量，如果手机上多装几个 IM 软件，每个月光心跳就好几兆流量没了，更不用说频繁的心跳带来的电量损耗。  
  
既然频繁心跳会带来耗电和耗流量的弊端，改进的方向自然是减少心跳频率，但也不能过于影响连接检测的实时性。基于这个需求，一般可以将心跳间隔根据程序状态进行调整，当程序在后台时(这里主要考虑安卓)，尽量拉长心跳间隔，5 分钟、甚至 10 分钟都可以。  
  
而当 App 在前台时则按照原来规则操作。连接可靠性的判断也可以放宽，避免一次心跳超时就认为连接无效的情况，使用错误积累，只在心跳超时 n 次后才判定当前连接不可用。当然还有一些小 trick 比如从收到的最后一个指令包进行心跳包周期计时而不是固定时间，这样也能够一定程度减少心跳次数。

## 文件中转模式

移动网络因为网络不稳定的客观因素存在，不适合实时发送较大的2进制文件（像电脑上的实时文件发送的那种），因为这涉及3方：客户端A、服务端、客户端B，任何两方的通讯因网络的不稳定而导致的重传等，都是个很不好处理的事情。  
  
现在多数情况下都是通过http先上传到中转文件服务器，成功后再通知接收方，这种情况下，因上传的过程只限于客户端A和服务端，只涉及两方，网络的不稳定，也只影响了发送者（而不涉及接收者，因为对方还不知道你正在发送文件呢），所以无论是从可靠性、复杂性，还是用户体验的处理上，这种方式都要简单的多。而这么多年的移动应用也证明，这种云中转暂存的方式是比较适合于当前的移动网络和移动应用体验的

## 服务器中转方式

几乎所有互联网IM产品都采用服务器中转这种方式进行消息传输，相对于P2P的方式，它有如下的优点:

* 能够支持更多P2P无法支持或支持不好的业务，如离线消息，群组，聊天室服务；
* 方便业务逻辑的拓展和新旧版本的兼容。

当然它也有自己的问题:服务器架构复杂，并发要求高。

# 未来性能方面

## 负载均衡 nginx （dubbo》》springcloud）

## 分片区存储 分区 分库分表

分区 分片机制即可。。。

可以按照年份稳定扩容

存储与消息中心都可以按照片区运行

# 安全性

防止信息泄露

远程命令安全性

性能安全

数据安全之片区隔离机制

消息加密 aes 端对端加密

# 附属的高级功能

## Sql的支持

# 附注：

## 表情包表达分类体系树

# Ref参考

XMPP入门\_编程之道，自强之路！-CSDN博客

移动端IM开发需要面对的技术问题-IM开发/专项技术区 - 即时通讯开发者社区!

HTTP文件上传服务器-支持超大文件HTTP断点续传的实现办法\_白袍小将的博客-CSDN博客

Atitit 音频 项目 系列功能表

聊聊四种即时通信(Instant Messaging)IM协议\_Stevin的技术博客的博客-CSDN博客.html

Atitit nlp自然语言识别功能体系树

webSocket进阶篇——STOMP Over Websocket - 简书

WebSocket子协议STOMP详解 - FEINIK的个人主页 - OSCHINA - 中文开源技术交流社区