“悦声”智能有声书

软件架构文档

版本 1.2

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 8/7/2019 | 1.0 | 第一次迭代 | 马包威 |
| 18/7/2019 | 1.1 | 第二次迭代，新增文本情感分析算法 | 马包威 |
| 1/8/2019 | 1.2 | 第四次迭代，新增推荐算法设计、修改逻辑视图 | 袁卓、马包威 |
|  |  |  |  |

目录

1. 简介 2

1.1 目的 2

1.2 参考资料 2

2. 用例视图 2

3. 逻辑视图 2

3.1 概述 2

3.2 在构架方面具有重要意义的设计包 2

4. 进程视图 2

5. 部署视图 2

6. 实现视图 2

7. 数据视图 2

8. 核心算法设计 2

软件架构文档 （简化版）

# 简介

## 目的

本文档将从构架方面对系统进行综合概述，其中会使用多种不同的构架视图来描述系统的各个方面。它用于记录并表述已对系统的构架方面作出的重要决策。

## 参考资料

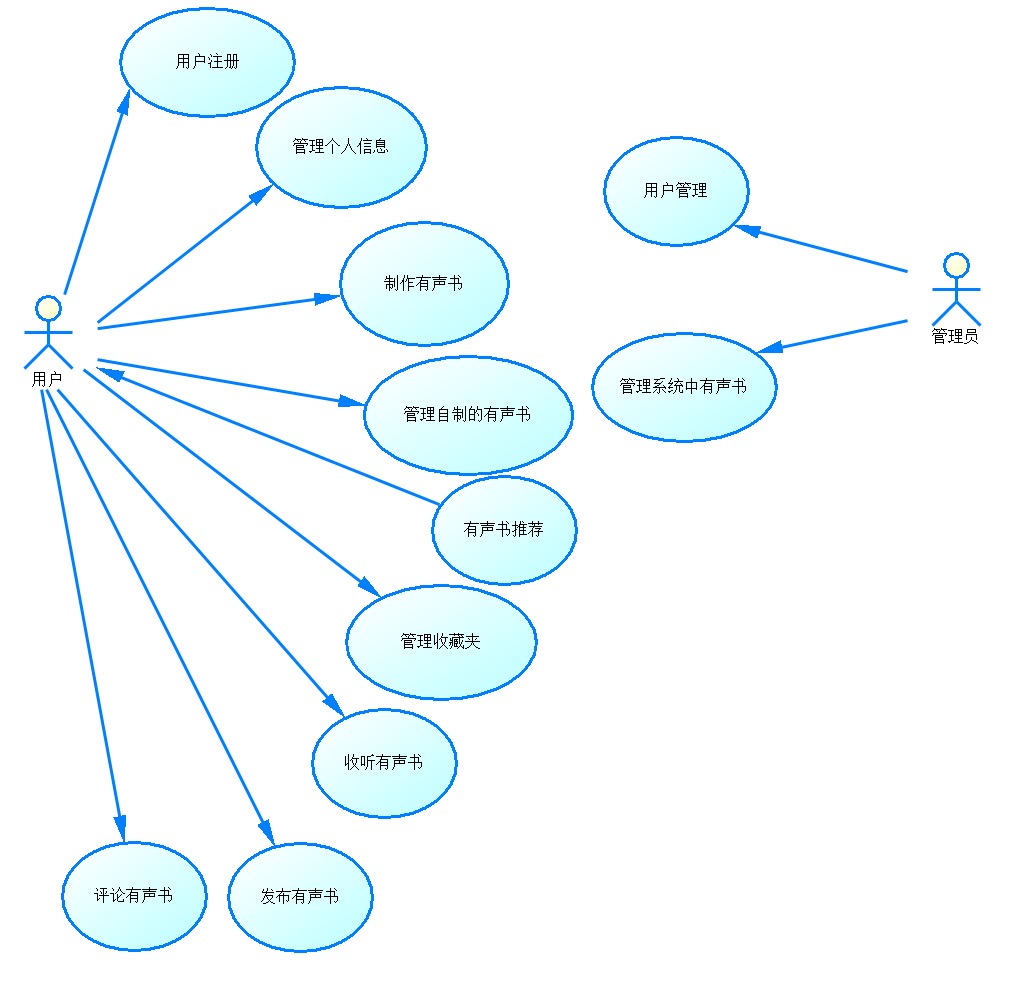
[1] 《“悦声”智能有声书软件需求规约》（马包威、周雪振、袁卓、王绍宇）

[2] 《软件工程原理》（沈备军，陈昊鹏，陈雨亭）高等教育出版社 2013.2 第一版

[3] 《中文情感词汇本体库》（林鸿飞 等）大连理工大学信息检索研究室

[4] 《hanLP：han Language Processing》 https://github.com/hankcs/HanLP

# 用例视图



# 逻辑视图

## 概述

本系统采用微服务架构风格，分为用户信息服务、有声书制作服务（包括输入文本制作有声书、录制音频制作有声书两种形式）、有声书收听服务、有声书推荐服务、有声书搜索服务、有声书管理服务这几个微服务模块。

## 在构架方面具有重要意义的设计包

### yuesheng.tv包和yuesheng.vt包中的Utility （有声书制作模块）

1. WordParser类：用于对文本进行分词操作
2. TimeTool类：用于获取当前的时间
3. SoundEffect类：用于音效的匹配
4. EmotionAnalysis类：用于分析文本的情感
5. CorrelationComputer类：用于背景音乐的匹配
6. Janitor类：用于删除多余的中间文件
7. FFMpeg类：用于对音频的处理。包括两个音频的混合、两个音频的连接、格式转换、音量调整、音频剪切。

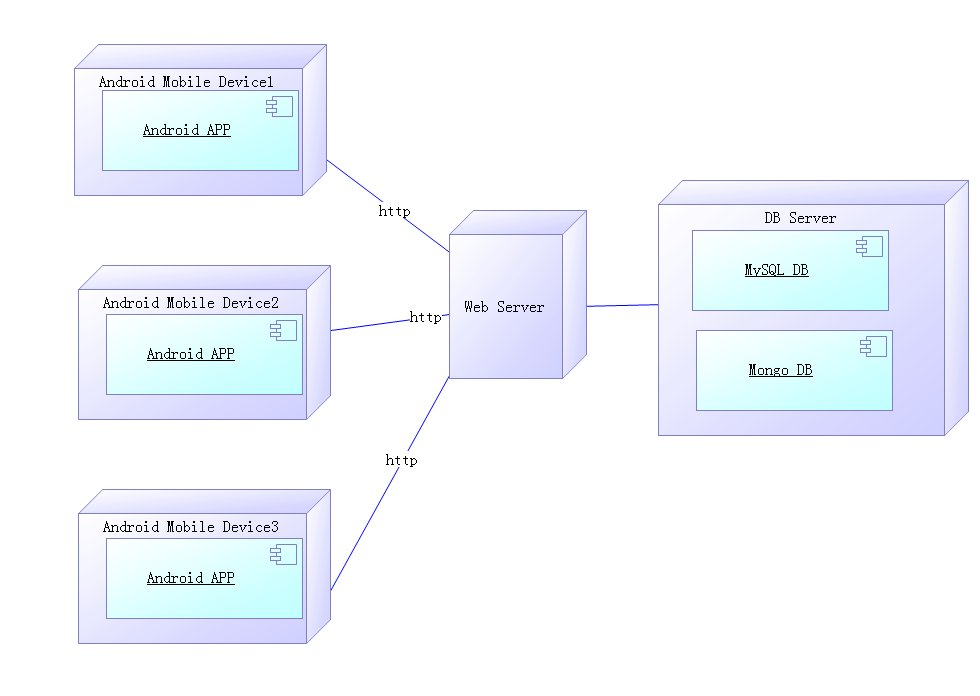
### 推荐系统模块

1. RecMap 类：生成相似度矩阵
2. RecList类：生成推荐列表

# 进程视图

说明：本系统中的多进程、多线程是由Spring框架负责管理。

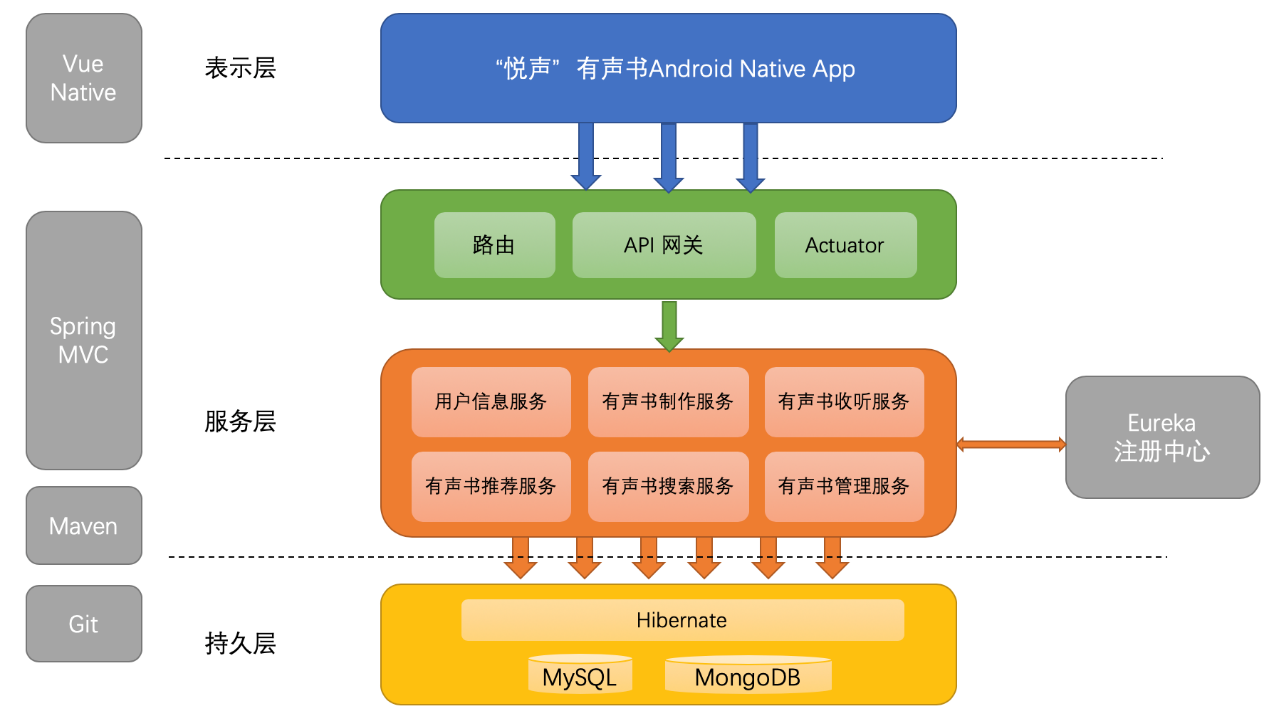
# 部署视图



# 实现视图

说明：本系统采用微服务架构实现：

1. 持久性层：使用关系型数据库MySQL和非关系型数据库MongoDB存储数据，建立应用系统的领域模型，实现对数据库的持久性操作。使用Hibernate对象—关系映射中间件。
2. 服务层：将服务分为用户信息服务、有声书制作服务、有声书收听服务、有声书推荐服务、有声书搜索服务、有声书管理服务这几个微服务，注册到Eureka，负责执行业务逻辑以处理用户请求，并调用数据访问层提供的持久性操作。
3. 表示层：负责向用户呈现界面，并接收用户请求发送给业务逻辑层。前端采用React native框架实现。



# 数据视图

## 数据库设计物理模型（CDM）



## 数据库设计概念模型（PDM）

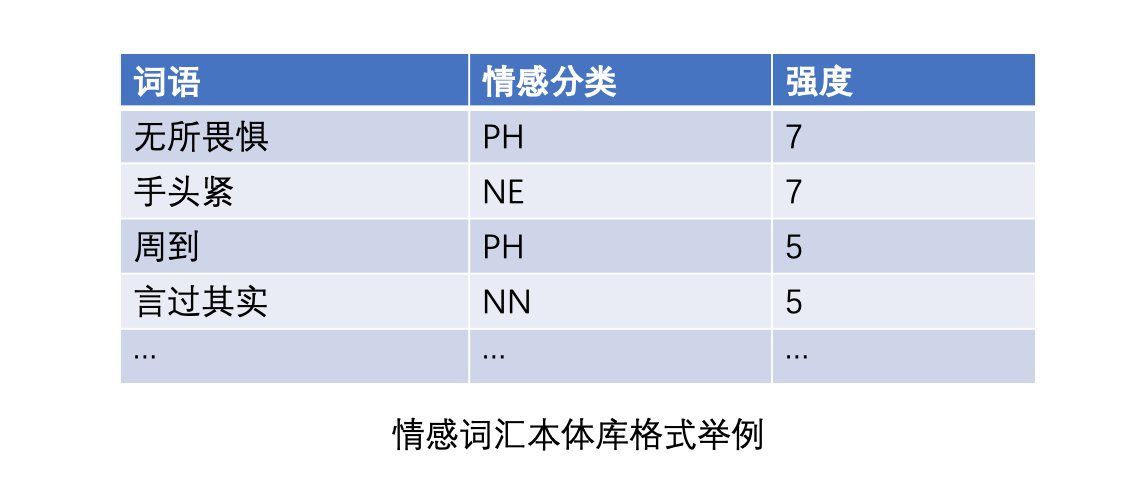


# 核心算法设计

## 文本情感分析和背景音乐匹配算法设计

本系统中，对于文本的情感分析及其背景音乐的选择匹配所采用的算法是：以分词为基础，利用情感词典对文本不同情感进行评估，并数据库中的背景音乐进行分类后建立音乐情绪分类与文本情感的关系映射，从而计算出音乐和文本的匹配程度。

该算法首先利用hanLP分词工具对文本进行分词处理，其中分词采用的词典为用户自定义，词典中包含了“中文情感词汇本体库”（资源来自大连理工大学信息检索研究室）中所有的情感词汇。情感词汇本体库格式如下表所示：

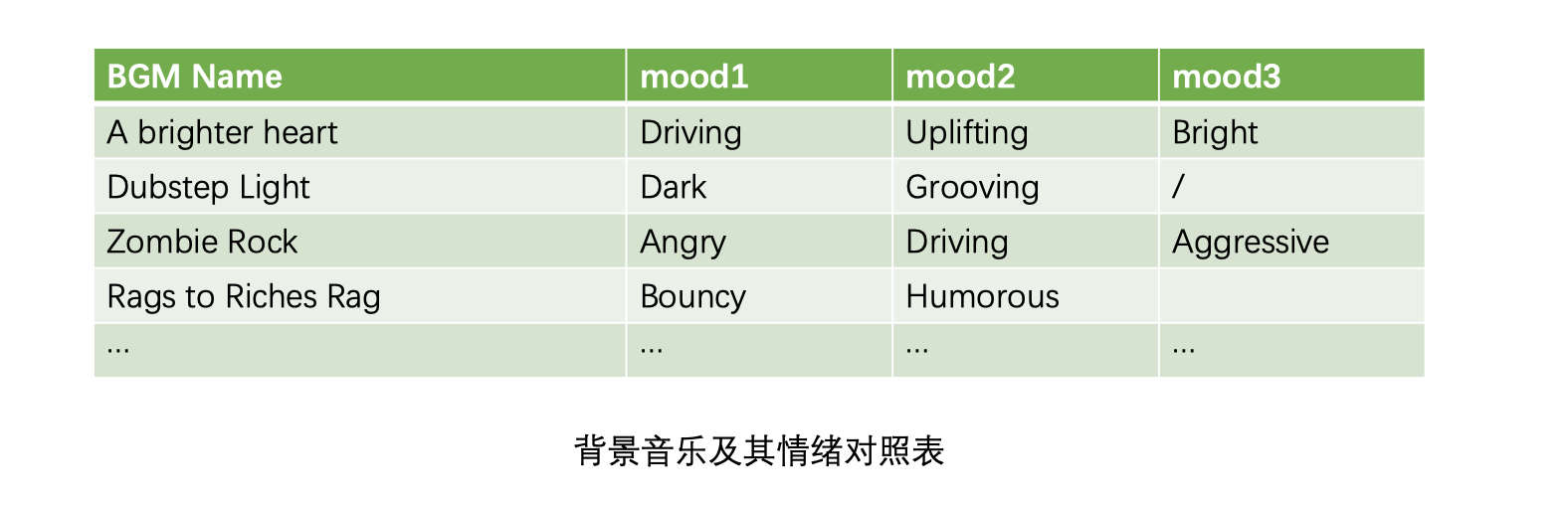


其中，情感分为7大类21小类。情感强度分为1,3,5,7,9五档，9表示强度最大，1为强度最小。情感分类如下表所示：

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 情感大类 | 情感类 | 例词 |
| 1 | 乐 | 快乐(PA) | 喜悦、欢喜、笑眯眯、欢天喜地 |
| 2 |  | 安心(PE) | 踏实、宽心、定心丸、问心无愧 |
| 3 | 好 | 尊敬(PD) | 恭敬、敬爱、毕恭毕敬、肃然起敬 |
| 4 |  | 赞扬(PH) | 英俊、优秀、通情达理、实事求是 |
| 5 |  | 相信(PG) | 信任、信赖、可靠、毋庸置疑 |
| 6 |  | 喜爱(PB) | 倾慕、宝贝、一见钟情、爱不释手 |
| 7 |  | 祝愿(PK) | 渴望、保佑、福寿绵长、万寿无疆 |
| 8 | 怒 | 愤怒(NA) | 气愤、恼火、大发雷霆、七窍生烟 |
| 9 | 哀 | 悲伤(NB) | 忧伤、悲苦、心如刀割、悲痛欲绝 |
| 10 |  | 失望(NJ) | 憾事、绝望、灰心丧气、心灰意冷 |
| 11 |  | 疚(NH) | 内疚、忏悔、过意不去、问心有愧 |
| 12 |  | 思(PF) | 思念、相思、牵肠挂肚、朝思暮想 |
| 13 | 惧 | 慌(NI) | 慌张、心慌、不知所措、手忙脚乱 |
| 14 |  | 恐惧(NC) | 胆怯、害怕、担惊受怕、胆颤心惊 |
| 15 |  | 羞(NG) | 害羞、害臊、面红耳赤、无地自容 |
| 16 | 恶 | 烦闷(NE) | 憋闷、烦躁、心烦意乱、自寻烦恼 |
| 17 |  | 憎恶(ND) | 反感、可耻、恨之入骨、深恶痛绝 |
| 18 |  | 贬责(NN) | 呆板、虚荣、杂乱无章、心狠手辣 |
| 19 |  | 妒忌(NK) | 眼红、吃醋、醋坛子、嫉贤妒能 |
| 20 |  | 怀疑(NL) | 多心、生疑、将信将疑、疑神疑鬼 |
| 21 | 惊 | 惊奇(PC) | 奇怪、奇迹、大吃一惊、瞠目结舌 |

将文本分词后，若词语出现在情感词典中，则文本对应的情感值得分将增加，增加值即为该情感词的强度。经过统计后，可计算得到该文本的21种情感得分值。

数据库中存储的背景音乐共有30余种情绪分类，每首背景音乐对应1~3种情绪。

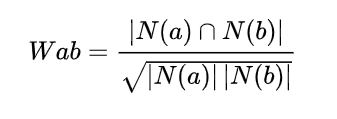


建立音乐情绪分类与文本情感的关系映射，关系度范围为-5至5，-5表示完全不相关，5表示完全相关。映射表如下所示。通过前述的文本情感得分和音乐情绪的关系度乘积相加，即可计算出音乐和文本的匹配程度，从而选取匹配度最高的背景音乐。

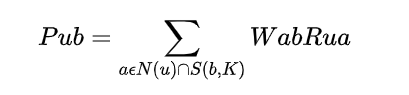
|  |  |  |
| --- | --- | --- |
| mood | sense | relevancy |
| Calming | PA | 4 |
| Calming | PE | 5 |
| Calming | PC | -1 |
| Calming | PD | 3 |
| Calming | PH | 3 |
| Calming | PG | 3 |
| Calming | PB | 4 |
| Calming | PK | 4 |
| Calming | NA | -5 |
| Calming | NB | 1 |
| Calming | NJ | -1 |
| Calming | NH | 1 |
| Calming | PF | 3 |
| Calming | NI | -5 |
| Calming | NC | -5 |
| Calming | NG | 2 |
| Calming | NE | -5 |
| Calming | ND | -5 |
| Calming | NN | -4 |
| Calming | NK | -3 |
| Calming | NL | -4 |
| Relaxing | PA | 4 |
| Relaxing | PE | 5 |
| Relaxing | PC | -2 |
| …… | …… | …… |
| Relaxing | NL | -4 |
| Uplifting | PA | 5 |
| …… | …… | …… |
| Uplifting | NL | -4 |
| …… | …… | …… |
| …… | …… | …… |

## 基于物品的协同过滤算法设计

基于物品的协同过滤就是根据用户历史选择物品的行为，通过物品间的相似度，给用户推荐其他物品。分为以下两个步骤。

1. 计算物品相似度，设喜欢物品 a 的用户数为 N(a)，喜欢物品 b 的用户数为 N(b)。则相似度计算公式为

即喜欢a物品的用户中，有多少比例的用户也喜欢b，比例越高，说明a与b的相似度越高。

1. 根据相似度进行推荐，假设N(u)是用户喜欢物品的集合，S(b, K)是和物品 b 最相似的 K 物品的集合，W(ab)是物品 a 和 b 的相似度，R(ua) 是用户 u 对物品 a 的兴趣。

实现了基本算法后，接下来解决推荐系统的三种冷启动问题。

1. 新系统冷启动。我们首先向系统中加入了一定量的用户和有声书，用户浏览轨迹来自于抽取的豆瓣用户的浏览轨迹。
2. 新用户冷启动。刚开始推荐最热门的有声书。
3. 新物品冷启动。在推荐列表中随机加入新物品。

其他边界条件：由推荐算法得到的列表不足设定数量，把热门有声书加入。

实际设定：

1. 对用户是否感兴趣有声书由评分获取，评分大于3分（总分为5分）即表示用户喜欢该有声书。
2. 相似度矩阵每隔一定时间更新一次，即假设在一段时间内协同过滤的条件不变，提高系统性能。