

中科院声学所在线识别引擎

概要设计说明

版 本 历 史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本/状态 | 作者 | 参与者 | 起止日期 | 备注 |
| V1.0 | 段志亮 |  | 2021-11-02 | 创建文件 |
|  |  |  |  |  |

目录

[1.概述 3](#_Toc86757367)

[**1.1简介** 3](#_Toc86757368)

[1.2 读者对象 4](#_Toc86757369)

[1.3术语定义 4](#_Toc86757370)

[2.总体设计 5](#_Toc86757371)

[2.1 总体需求 5](#_Toc86757372)

[2.2 运行环境 6](#_Toc86757373)

[**2.2.1 软件环境 6**](#_Toc86757374)

[**2.2.2 硬件环境 6**](#_Toc86757375)

[2.3系统总体结构设计 7](#_Toc86757376)

[**2.3.1 系统结构说明 7**](#_Toc86757377)

[**2.3.2系统总体时序流程 8**](#_Toc86757378)

[3.识别服务模块设计 11](#_Toc86757379)

[3.1 数据接收及发送模块 11](#_Toc86757380)

[3.2 数据解析及封装模块 12](#_Toc86757381)

[3.3 会话流程处理模块 13](#_Toc86757382)

[4.数据设计 15](#_Toc86757383)

[4.1 会话开始输入、输出数据 15](#_Toc86757384)

[4.2 会话交互输入、输出数据 16](#_Toc86757385)

[4.3 获取会话结果输入、输出数据 19](#_Toc86757386)

[4.4 会话结束输入、输出参数 20](#_Toc86757387)

[5.配置及测试设计 22](#_Toc86757388)

[5.1 配置文件设计 22](#_Toc86757389)

[5.2 测试功能点 24](#_Toc86757390)

[6.可靠性设计 26](#_Toc86757391)

[6.1 健壮性设计 26](#_Toc86757392)

[6.2 故障恢复 27](#_Toc86757393)

[7.扩展性设计 27](#_Toc86757394)

[8.安全性设计 29](#_Toc86757395)

[9.维护性设计 30](#_Toc86757396)

[10.易用性设计 31](#_Toc86757397)

# 1.概述

1

**1.1简介**

在线语音识别系统（Automatic Speech Recognition）是以语音为处理对象，通过语音信号处理技术，让机器把语音转换为相应的文本的系统。实现系统具备识别率高、性能稳定、接入方便等特点；支持一句话识别和实时识别两种服务，能够满足不同业务类型需求。

实时识别是对音频流进行识别，适用于实时性要求的场景。

一句话识别是对20s内的短语音进行识别，适用于短语音转写的场景。

语言和方言：目前支持中文普通话和带有一定方言的中文普通话；

音频属性：支持wav、pcm引擎格式，支持8k、16k采样率的单声道语音；

音频数据长度：实时识别建议每个数据包为1600字节，一句话识别建议每包大小不超过100k。

## 1.2 读者对象

本文档的读者对象为在线识别系统的开发人员、测试人员、系统维护人员及接入识别系统的第三方业务人员，通过本文档能够从总体上了解识别系统的架构形式及数据流向。

本说明给出在线识别系统的设计说明，包括最终实现的系统必须满足的功能、性能、接口、附属测试工具程序及设计约束等。

目的在于：

* 为开发人员提供依据；
* 为代码修改、维护提供条件；

## 1.3术语定义

语音识别（Automatic Speech Recognition），简称（ASR）。

# 2.总体设计

2

## 2.1 总体需求

功能需求：满足8k16bit pcm、16k16bit pcm语音转文字功能；

性能需求：语音识别出对应文本内容延时不超过200ms；

接口需求：接口参数简单明了，接口中的每个参数都要有实际意义，保证接口调用流程清晰。

系统验证工具：提供系统完整性，功能正确性的验证工具；

## 2.2 运行环境

### 2.2.1 软件环境

|  |  |  |
| --- | --- | --- |
| 分类 | 名称 | 版本 |
| 操作系统 | Centos | 7.0以上 |
| 数据库 | Redis | 3.2.0 |
| 能力接口 | Tomcat | 9.0.35 |

### 2.2.2 硬件环境

|  |  |  |
| --- | --- | --- |
| 服务器 | 最低配置 | 推荐配置 |
| redis服务器 | CPU：1  内存：256M | CPU：1  内存：1G |
| 识别服务器 | CPU：4  内存：10G  磁盘：20G | CPU：10  内存：15G  磁盘：30G |

## 2.3系统总体结构设计

### 2.3.1 系统结构说明

在线识别系统服务系统主要分为：分布式转发服务、能力接口服务、会话缓存服务、识别引擎服务四个子系统。

分布式转发服务：对外提供调用端口和url，转发客户端请求到不同的能力接口；

能力接口服务：接收nginx转发请求，校验数据有效性及封装成识别服务需要的数据格式；

会话缓存服务：redis缓存能力接口发送过来的会话信息及识别服务存放的服务信息；

识别服务：实现语音转文字功能。

系统总体流程图如下所示：



### 2.3.2系统总体时序流程

能力接口、redis和识别服务构建生产者-消费者模式，降低了能力接口和识别服务耦合度，极大的提高了识别服务的横向扩展能力。

系统总体时序流程如下所示：

 从上面时序图上看出，能力接口将新的会话信息存放在redis，空闲的识别服务获取会话信息后，将本身信息写入redis，能力接口根据会话Id从redis中读取识别服务信息，之后能力接口和识别服务才建立一对一的连接。该设计方式，使识别服务的扩容更加便利。

# 3.识别服务模块设计

3

识别服务包含接收数据模块、解析及封装数据模块和流程控制模块。

模块间关系如下：



## 3.1 数据接收及发送模块

识别服务和tomcat之间通过socket通信，数据接收及发送模块负责接收tomcat发送过来的请求数据并将处理后的数据返回给tomcat；

处理流程如下：



**相关函数介绍：**

**void** **CallBackFunc\_Server**(std::string &out\_action, std::string &out\_msg, **const** std::string &\_in\_action, **const** std::string &\_in\_msg)

函数功能：该函数主要接收能力接口端发送的请求数据，并将识别后的数据返回给能力接口。

## 3.2 数据解析及封装模块

识别服务需要将接收到的请求数据进行解析处理，tomcat发送到识别服务端是数据形式为json字符串（详见数据设计），识别服务将json字符串解析为本地方便使用的结构体形式；

识别服务返回数据到tomcat需要按照设计的格式进行返回，需要使用数据的封装模块，将返回数据进行整理封装，然后通过socket方式返回到tomcat端。

相关函数介绍：

**bool** **deserialize**(redis\_param\* item, **const** **char**\* buffer, size\_t size)

函数功能：解析接收到的数据为服务方便使用的结构形式。

std::string **getXmlStr**(ACand\* cand, asr\_resp\_t res\_data, **int** selectNum, **char**\* encoding, **int** realTimeContent,**int** soundType)

函数功能：封装返回的识别数据。

## 3.3 会话流程处理模块

会话流程处理是识别服务的核心模块，负责从会话开始到结束的整个流程控制；

会话开始从redis中获取会话信息，将当前服务的信息存入redis；tomcat获取到服务信息后，通过socket方式将语音流数据发送到识别服务，识别服务调用引擎接口对语音进行识别处理，语音发送结束后，返回识别结果信息，结束会话。

会话控制流程图如下：



相关函数介绍：

static void \*DecodeProcThread(void \*parameter)

函数功能：该函数主要实现数据处理工程的逻辑控制功能；

# 4.数据设计

4

## 4.1 会话开始输入、输出数据

会话开始识别引擎从redis数据库中获取会话信息数据，解析数据，根据数据参数确定语音数据格式，然后将本机的ip和端口信息写入redis中。

会话开始输入数据：

|  |  |  |
| --- | --- | --- |
| 参数 | 类型 | 说明 |
| aue | 字符串 | 非编码原始语音 |
| auf | 字符串 | 语音采样率 |
| cmd | 字符串 | 会话状态 |
| sid | 字符串 | 会话Id |

如下实例：

{  
    **"request"**:{  
        **"aue"**:**"raw"**,  
        **"auf"**:**"audio/L16;rate=8000"**,  
        **"cmd"**:**"ssb"**  
    },  
    **"sid"**:**"16e374f04d8348e487dab492552537a8"**  
}

会话开始输出数据：

|  |  |  |
| --- | --- | --- |
| 参数 | 类型 | 说明 |
| ret | 整型 | 响应状态 |
| sid | 字符串 | 会话Id |
| addr | 字符串 | 识别服务地址信息 |

**实例如下：**

{  
    **"response"**:{  
        **"ret"**:**0**  
    },  
    **"sid"**:**"16e374f04d8348e487dab492552537a8"**,  
    **"addr"**:**"192.168.0.52:10200"**  
}

## 4.2 会话交互输入、输出数据

会话的交互阶段主要是语音数据流和识别信息的传递，能力接口将客户端发送的语音数据流发送到识别服务，识别服务将对语音数据进行识别，并返回识别信息。

|  |  |  |
| --- | --- | --- |
| 参数 | 类型 | 说明 |
| data | 字符串 | Base64加密的语音数据 |
| cmd | 字符串 | 会话状态 |
| syncid | 字符串 | 语音包序号 |
| sid | 字符串 | 会话Id |

实例如下：

{  
    **"request"**:{  
        **"data"**:**""**,  
        **"cmd"**:**"auw"**,  
        **"syncid"**:**"1"**  
    },  
    **"sid"**:**"16e374f04d8348e487dab492552537a8"**  
}

输出数据：

|  |  |  |
| --- | --- | --- |
| 参数 | 类型 | 说明 |
| sid | 字符串 | 会话Id |
| recStatus | 字符串 | 识别状态 |
| engine\_name | 字符串 | 引擎地址 |
| result | json字符串 | 识别结果 |
| bg | 字符串 | 检测到有效语音 |
| ed | 字符串 | 检测到有效语音尾点 |

**实例如下：**

<?xml version="1.0" encoding="utf-8"?>

<results>

<sid>**16e374f04d8348e487dab492552537a8**</sid>

<recStatus>5</recStatus>

<engine\_name>192.168.0.52</engine\_name>

<result>[{ "text": "你好。"]</result>

<bg>1</bg>

<ed>0</ed>

</results>

## 4.3 获取会话结果输入、输出数据

获取会话识别结果阶段是在客户端将语音发送完毕后，通过调用能力接口获取识别结果信息。

输入参数：

|  |  |  |
| --- | --- | --- |
| 参数 | 类型 | 说明 |
| cmd | 字符串 | 会话状态 |
| syncid | 字符串 | 序号 |
| sid | 字符串 | 会话Id |

**实例如下：**

{  
    **"request"**:{  
        **"cmd"**:**"grs"**,  
        **"syncid"**:**"1306"**  
    },  
    **"sid"**:**"16e374f04d8348e487dab492552537a8"**  
}

输出参数：

|  |  |  |
| --- | --- | --- |
| 参数 | 类型 | 说明 |
| sid | 字符串 | 会话Id |
| recStatus | 字符串 | 识别状态 |
| engine\_name | 字符串 | 引擎地址 |
| result | json字符串 | 识别结果 |
| bg | 字符串 | 检测到有效语音 |
| ed | 字符串 | 检测到有效语音尾点 |

**实例如下：**

<?xml version="1.0" encoding="utf-8"?>

<results>

<sid>**16e374f04d8348e487dab492552537a8**</sid>

<recStatus>5</recStatus>

<engine\_name>192.168.0.52</engine\_name>

<result>[{ "text": "你好。", "phoneme": "你好。", "segtime": "0.31 0.86", "score": 1.0 }]</result>

<bg>1</bg>

<ed>1</ed>

</results>

## 4.4 会话结束输入、输出参数

会话结束是在客户端获取到识别结果或者服务端返回异常信息时，客户端调用能力接口通知服务端结束会话。

输入参数：

|  |  |  |
| --- | --- | --- |
| 参数 | 类型 | 说明 |
| cmd | 字符串 | 会话状态 |
| sid | 字符串 | 会话Id |

实例如下：

{  
    **"request"**:{  
        **"cmd"**:**"sse"**  
    },  
    **"sid"**:**"16e374f04d8348e487dab492552537a8"**  
}

输出参数：

|  |  |  |
| --- | --- | --- |
| 参数 | 类型 | 说明 |
| ret | 整型 | 会话结束状态 |
| sid | 字符串 | 会话Id |

**实例如下：**

{  
    **"ret"**:0,  
    **"sid"**:**"16e374f04d8348e487dab492552537a8"**  
}

# 5.配置及测试设计

5

## 5.1 配置文件设计

设置配置文件，方便使用不同控制逻辑，支持不同的功能及结果输出，识别服务配置文件实现如下目的：

* 通过参数控制支持特殊功能
* 检测异常数据
* 日志输出
* 引擎资源加载
* 数据库连接
* 服务地址配置

配置文件主要参数如下表：

|  |  |  |
| --- | --- | --- |
| 参数 | 取值 | 说明 |
| SaveVoice | 范围0－3 | 0代表不保存语音；1代表只保存接收到的语音，2代表只保存转换后的pcm16语音；3代表同时保存语音和pcm16语音 |
| VoiceDir | ../voice | 保存语音路径 |
| checkSyncid | 范围0－1 | 是否检查语音包序号，0：不检查，1：检查 |
| WaitResultTime | 范围1－1000 | 从接收到数据结束到识别结果出来的等待时间，单位0.1秒； |
| NetTimeout | 范围>=100000 | 接收或发送数据的网络超时时间，单位为us |
| RedisPasswdOn | 范围0－1 | 设置redis集群是否设置密码，1：设置，0：不设置 |
| RedisCluster | ip:端口形式 | Redis服务地址 |
| RedisPasswd | 字符串 | Redis密码 |
| LocalIP | 字符串 | 服务ip |
| SaveRegResult | 范围0－1 | 是否保存语音识别结果，0不保存，非0保存 |
| RegResultPath | ../txtFile | 保存结果路径 |
| RedisKey | 字符串 | Redis中会话队列key值 |
| IsRealTimeModel | 范围0－1 | 0:一句话识别，1：实时识别 |
| RealtimeResultModel | 范围0－2 | 实时识别时返回的结果形式：0：返回断句结果(含置信度)，1：返回开始到断句所有结果(引擎vad打开)，2：返回断句结果(不含置信度) |
| LogPath | ../log | 日志路径 |
| LogLevel | 范围0－8 | 日志级别 |
| LogSize | 10000 | 输出日志文件大小 |

## 5.2 测试功能点

根据设计需求，对服务必须满足的功能点进行验证测试。

|  |  |
| --- | --- |
| 测试功能 | 期望结果 |
| 服务启动验证 | 正确配置服务参数，启动服务，CPU、内存满足情况下，服务能够正常启动。 |
| 8k16bit pcm语音 | 参数传递正确情况下，输出识别结果。 |
| 16k16bit pcm语音 | 参数传递正确情况下，输出识别结果。 |
| 保存语音 | 配置保存语音参数，保存语音正常。 |
| 关闭保存语音 | 配置不保存语音参数，不保存语音。 |
| 保存识别结果 | 配置保存识别结果参数，保存识别结果正常。 |
| 关闭保存识别结果 | 配置不保存识别结果参数，不保存识别结果 |
| 检查数据包序号 | 配置检查数据包序号，数据包乱序返回错误信息 |
| 不检查数据包序号 | 配置不检查数据包序号，数据包乱序，不返回错误信息 |
| 单句识别 | 参数传递正确情况下，输出识别结果。 |
| 实时识别 | 参数传递正确情况下，输出识别结果。 |
| 实时识别，配置不同返回结果 | 返回不同的结果形式 |
| 日志级别 | 调整不同的日志级别，日志输出内容不同 |
| 日志文件大小 | 调整日志文件大小，保存日志文件大小不同 |

# 6.可靠性设计

6

## 6.1 健壮性设计

服务运行过程中，客户端发送过来的异常数据及异常会话可能导致输出结果错误、服务崩溃等问题，为避免此类问题软件设计中有如下措施：

* 检查输入数据的数据类型,防止异常数据引起输出结果错误。
* 模块调用时检查参数的合法性,对非法参数请求返回错误信息提示。
* 简化软件的功能复杂性,与服务无关功能禁止加在服务上。
* 网络异常超时，返回对应错误信息。
* 识别超时等内部异常错误，返回错误码信息

如下是部分错误码信息：

|  |  |
| --- | --- |
| 错误码 | 说明 |
| 52 | 会话已经结束，会话不存在 |
| 66 | 等待识别响应超时 |
| 82 | 请求数据包乱序 |
| 97 | 请求语音包太小 |
| 98 | 解析请求数据失败 |

## 6.2 故障恢复

服务运行中异常情况导致的服务重启，识别服务需要不在人为干预的情况下自动重启。

# 7.扩展性设计

7

会话通过生产者消费者模式处理，降低能力接口和识别服务的耦合度，能力接口接收到会话后，将会话信息存放redis，识别服务端有空闲线程时，主动从redis队列中获取会话信息，通过redis队列减弱能力接口和识别服务的耦合关系，实现能力接口和识别服务的多对多关系。

如下图所示：



能力接口和识别服务的低耦合度，可以使新增接点的识别服务方便的实现横向扩展，满足后期服务的大并发需求。

# 8.安全性设计

8

从客户端发送的语音数据需要进行base64加密处理，同时从服务端返回的响应数据也进行base64加密，在传输的过程中一定程度上保证数据的安全性。

# 9.维护性设计

9

能力接口和识别服务之间通过redis建立对应关系，能力接口、redis、识别服务三者构成的生产者消费者模式，使新的识别服务接入识别系统时，不需要关闭识别系统，从而实现识别服务的横向扩容。

识别服务的参数配置在启动服务时默认使用最优配置，如果机器环境（如cpu、内存）发送变化时，根据需要停止服务重新配置。

# 10.易用性设计

10

代码编译：在代码工程指定目录下执行编译脚本即可生成可执行文件；将可执行文件替换到部署系统的指定目录下即可实现系统更新；

系统部署：提供完整的文档介绍，用户可以根据文档说明完成系统部署启动；

系统调用：系统提供了调用demo，用户可以先使用demo测试验证系统的正确性，然后可以根据接口文档和demo完成接口集成调用。