

中科院声学所mrcp

详细设计说明

版 本 历 史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本/状态 | 作者 | 参与者 | 起止日期 | 备注 |
| V1.0 | 段志亮 |  | 2021-11-25 | 创建文件 |
|  |  |  |  |  |

目录

[1.概述 3](#_Toc88816818)

[1.1编写目的 3](#_Toc88816819)

[1.2 读者对象 3](#_Toc88816820)

[1.3术语定义 4](#_Toc88816821)

[2.总体设计 4](#_Toc88816822)

[2.1 总体需求 4](#_Toc88816823)

[2.2 运行环境 5](#_Toc88816824)

[2.2.1 软件环境 5](#_Toc88816825)

[2.2.2 硬件环境 5](#_Toc88816826)

[2.3服务总体结构设计 5](#_Toc88816827)

[2.3.1 mrcp识别服务总体逻辑 5](#_Toc88816828)

[2.3.2服务总体时序流程 6](#_Toc88816829)

[3.数据设计 8](#_Toc88816830)

[3.1 与ASR交互数据结构 8](#_Toc88816831)

[3.1.1 会话开始输入、输出数据 8](#_Toc88816832)

[3.1.2 会话交互输入、输出数据 9](#_Toc88816833)

[3.1.3 获取会话结果输入、输出数据 12](#_Toc88816834)

[3.1.4 会话结束输入、输出参数 13](#_Toc88816835)

[3.2 模块内数据结构 14](#_Toc88816836)

[3.2.1 接收客户端请求数据 14](#_Toc88816837)

[3.2.2 数据封装数据结构 16](#_Toc88816838)

# 1.概述

1

1.1编写目的

本文档主要用于为实现系统功能而进行的系统数据设计说明，具体描述了系统包含的的软件模块的实现流程、功能、接口、数据结构等内容，供项目组开发人员和软件维护人员阅读。

## 1.2 读者对象

本文档的读者对象为mrcp服务的开发人员、测试人员、系统维护人员，通过本文档能够从总体上了解mrcp识别服务的数据流向。

本说明给出mrcp服务的设计说明。

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：接收客户端语音流，发送到ASR识别服务端，并给客户端返回语音流的识别结果；

性能需求：支持处理并发任务；

## 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 |
| 编译环境 | gcc/g++ | 4.8.5 |

### 2.2.2 硬件环境

|  |  |  |
| --- | --- | --- |
| 服务器 | 最低配置 | 推荐配置 |
| mrcp服务器 | CPU：2  内存：10G  磁盘：20G | CPU：4  内存：15G  磁盘：50G |

## 2.3服务总体结构设计

### 2.3.1 mrcp识别服务总体逻辑

mrcp识别服务总体逻辑图如下：



### 2.3.2服务总体时序流程

服务总体时序流程如下所示：



# 3.数据设计

3

## 3.1 与ASR交互数据结构

### 3.1.1 会话开始输入、输出数据

会话开始输出数据：

|  |  |  |
| --- | --- | --- |
| 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"

}

会话开始输入数据：

|  |  |  |
| --- | --- | --- |
| 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"**  
}

### 3.1.2 会话交互输入、输出数据

输出数据：

|  |  |  |
| --- | --- | --- |
| 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"**  
}

输入数据：

|  |  |  |
| --- | --- | --- |
| 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>

### 3.1.3 获取会话结果输入、输出数据

输出参数：

|  |  |  |
| --- | --- | --- |
| 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"**  
}

输入参数：

|  |  |  |
| --- | --- | --- |
| 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>

### 3.1.4 会话结束输入、输出参数

输出参数：

|  |  |  |
| --- | --- | --- |
| 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"**  
}

输入参数：

|  |  |  |
| --- | --- | --- |
| 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"**  
}

## 3.2 模块内数据结构

### 3.2.1 接收客户端请求数据

**数据结构如下：**

**typedef** **struct** AsrRecogMsg

{

RecogMsgType type;

mrcp\_engine\_channel\_t \*channel;

mrcp\_message\_t \*request;

} AsrRecogMsg;

|  |  |  |
| --- | --- | --- |
| **结构体参数** | **类型** | **说明** |
| type | 枚举值 | 请求消息命令 |
| channel | 指针 | 请求会话通道 |
| request | 指针 | 请求消息 |

关联结构数据

**typedef** **enum**

{

*DEMO\_RECOG\_MSG\_OPEN\_CHANNEL*,

*DEMO\_RECOG\_MSG\_CLOSE\_CHANNEL*,

*DEMO\_RECOG\_MSG\_REQUEST\_PROCESS*

} RecogMsgType;

相关代码逻辑如下：；

AsrRecogMsg \*recog\_msg = (AsrRecogMsg\*) msg->data;

apt\_log(APT\_LOG\_MARK, *APT\_PRIO\_INFO*, "=RecogChannelMsgProces = [%d]", recog\_msg->type);

**switch** (recog\_msg->type) // 0,2

{

**case** *DEMO\_RECOG\_MSG\_OPEN\_CHANNEL*:

mrcp\_engine\_channel\_open\_respond(recog\_msg->channel, TRUE); /\* open channel and send asynch response \*/

**break**;

**case** *DEMO\_RECOG\_MSG\_CLOSE\_CHANNEL*:

{

AsrRecogChannel \*recog\_channel = (AsrRecogChannel \*) recog\_msg->channel->method\_obj;

apt\_log(APT\_LOG\_MARK, *APT\_PRIO\_INFO*, "==RecogChannelMsgProcess = [%d],%s", recog\_msg->type,recog\_channel->channel->id.buf);

**if** (recog\_channel->audioFile\_)

{

WriteWavHeaderAndClose(recog\_channel->audioFile\_);

recog\_channel->audioFile\_ = NULL;

}

mrcp\_engine\_channel\_close\_respond(recog\_msg->channel);

**break**;

}

**case** *DEMO\_RECOG\_MSG\_REQUEST\_PROCESS*:

RecogMrcpRequestDispatch(recog\_msg->channel, recog\_msg->request);

**break**;

**default**:

**break**;

}

### 3.2.2 数据封装数据结构

返回给客户端数据是xml形式，结构如下：

<?xml version="1.0" encoding="gb2312"?>

<result>

<interpretation grammar="session:1234567abcdef" confidence="0">

<instance>你好</instance>

<input mode="speech">你好;472e87f7c23c45bespeechrecog</input>

</interpretation>

</result>

|  |  |  |
| --- | --- | --- |
| Xml参数 | 类型 | 说明 |
| grammar | 字符串 | 语法 |
| instance | 字符串 | 识别结果 |
| input | 字符串 | 输入说明 |

数据封装主要代码逻辑如下：

xml.AddElem("result");

xml.IntoElem();

xml.AddElem("interpretation");

xml.SetAttrib("grammar", asr->grammarID\_);

xml.SetAttrib("confidence", asr->httpClient\_->GetConfidence().c\_str());

string speech\_id\_temp;

**if**(g\_asrSrvConfig.xmlResultByUtf8)

speech\_id\_temp = text;

**else**

speech\_id\_temp = util::ConvertUtf8ToGBK(text);

xml.AddChildElem("instance", speech\_id\_temp.c\_str());

speech\_id\_temp += ";";

speech\_id\_temp += MRCP\_MESSAGE\_ID(asr->recogRequest\_);

speech\_id\_temp += MRCP\_MESSAGE\_NAME(asr->recogRequest\_);

xml.AddChildElem("input", speech\_id\_temp.c\_str());

xml.SetChildAttrib("mode", "speech");