软件概要设计说明书

Software Concept Design Document

版本历史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Date | Version | Description | Author |
| 1 | 2014.6.12 | V0.01 | 初始版本 | Zhonglong.Chen |
|  |  |  |  |  |
|  |  |  |  |  |

目录

[1引言 3](#_Toc390883069)

[1.1背景 3](#_Toc390883070)

[1.2基线 3](#_Toc390883071)

[1.3范围 3](#_Toc390883072)

[1.4定义 3](#_Toc390883073)

[1.5参考资料 3](#_Toc390883074)

[2总体设计 3](#_Toc390883075)

[2.1概述 3](#_Toc390883076)

[2.1.1需求规定 3](#_Toc390883077)

[2.1.2 运行环境 4](#_Toc390883078)

[2.1.3 拟复用的已有函数库 4](#_Toc390883079)

[2.2设计 4](#_Toc390883080)

[2.2.1设计构思 4](#_Toc390883081)

[2.2.2系统流程 (画出动态运行流程) 5](#_Toc390883082)

[2.2.3结构设计 8](#_Toc390883083)

[2.2.4功能需求与程序的关系 (具体需求与程序模块的对应关系) 9](#_Toc390883084)

[3系统数据结构设计 11](#_Toc390883085)

[3.1数据结构 11](#_Toc390883086)

[4接口设计 11](#_Toc390883087)

[4.1用户接口 11](#_Toc390883088)

[4.2 外部接口 (与外部程序系统之间交互的接口) 11](#_Toc390883089)

[4.2.1 语音引擎接口 11](#_Toc390883090)

[4.2.2 应用间通信接口 12](#_Toc390883091)

[4.3 内部接口(内部组件之间调用的接口) 12](#_Toc390883092)

[4.3.1 Domain接口 12](#_Toc390883093)

[4.3.2 DomainManager接口 12](#_Toc390883094)

[4.3.3 ContextUtility接口 12](#_Toc390883095)

[5运行设计 13](#_Toc390883096)

[5.1 系统初始化（初始化流程图） 13](#_Toc390883097)

[5.2 运行模块组合 13](#_Toc390883098)

[5.3 运行时间 13](#_Toc390883099)

[6出错处理和维护 13](#_Toc390883100)

[6.1 出错信息 13](#_Toc390883101)

[6.2 补救措施 13](#_Toc390883102)

[6.3 系统维护 13](#_Toc390883103)

[7尚待解决的问题 13](#_Toc390883104)

# 1引言

## 1.1背景

本文是Flagship 2k15机种Voice应用的概要设计文档，是Voice详细设计和编码的指导文件。

## 1.2基线

规格基线：<http://172.16.144.44/CN_Flagship/Spec/0.Baseline/V1.0/Voice/2K15_Voice&NLU&Vocalizer_FRS.xlsx> #158

## 1.3范围

## 1.4定义

## 1.5参考资料

1. Voice NLU 识别结果接口说明.docx
2. mediacloud-voice-百度语义理解开放平台意图表示协议说明书.pdf

# 2总体设计

## 2.1概述

### 2.1.1需求规定

Voice应用包括Voice Control（语音控制）和NLU & Vocalizer（自然语言处理）两部分，用于识别和处理用户的语音输入。按引擎和UX的规格，整个Voice操作被设计成一系列独立的Domain，每个Domain完成特定场景的控制，如电视、浏览器、天气等。

Voice应用同时包含声纹登录功能，但不包含文本输入（文本输入在输入法应用中实现）。

### 2.1.2 运行环境

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 平台 | 操作系统 | 屏幕分辨率 | 存储空间占用 | 内存空间占用 | CPU占用 |
| *待定* | *Android 4.2+* | *1920x1080* | *<= 15M* | *<= 30M* | *<= 15%* |

### 2.1.3 拟复用的已有函数库

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 函数库名称 | 来源 | 预计节约工时（人力） |
| 1 |  |  |  |
| 2 |  |  |  |

没有可复用函数库，因为这是个全新的应用和交互方式，在以前机种上从未开发过。

## 2.2设计

### 2.2.1设计构思

Voice应用采用TUF框架，以实现统一的界面风格和更换主题功能。

Voice应用采用Float Window Manager框架，仅需要一个Service和一个FloatActivity对象，所有Domain在UI上具体体现为一个个View（图1红色框内部分），加入到FloatActivity对象中。

每个Domain的View一般包含三项内容，用户语音识别出来的文本（InTextView），对用户问题的回答（OutTextView），以及相关的附加视图（AttachView）。以Weather场景为例，如下图：

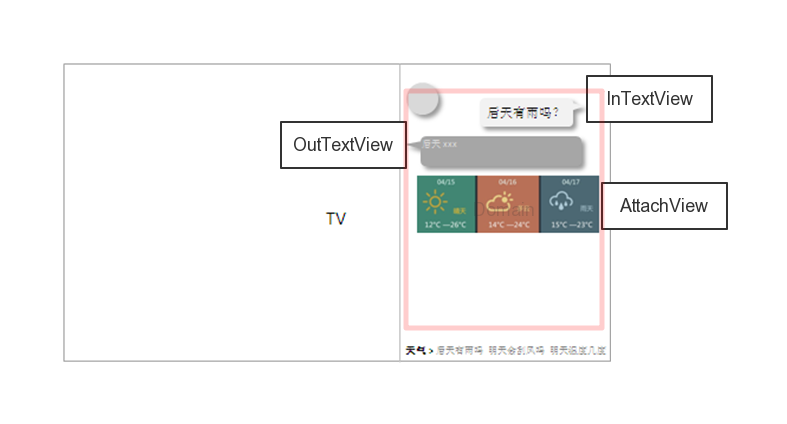


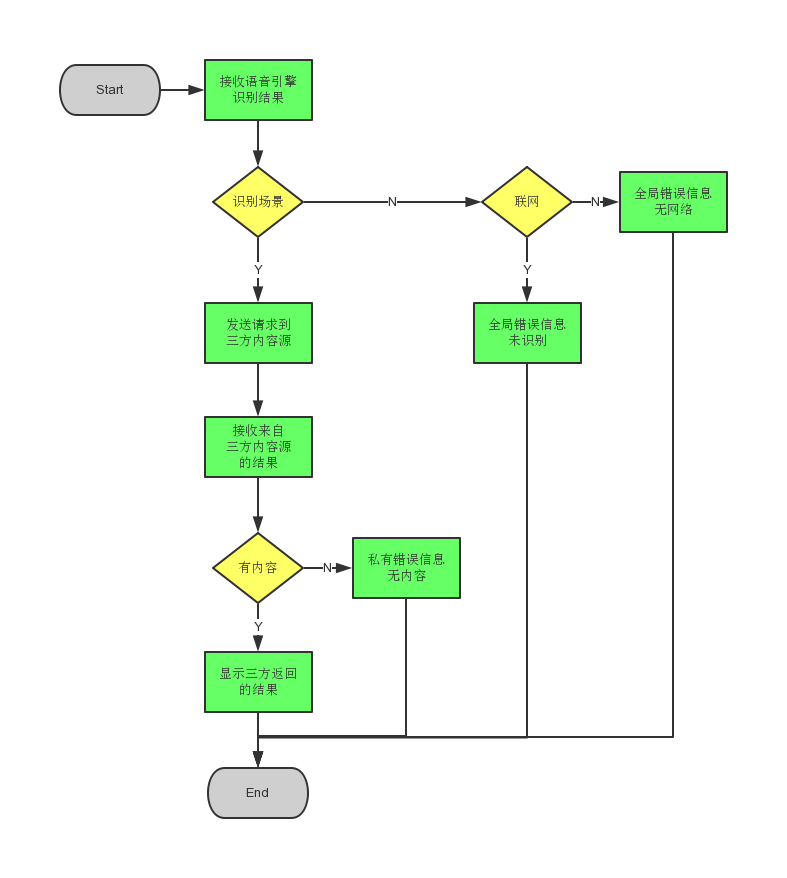
图1 场景示意图

一般来说这三项内容只需要至少有一项，而且按场景定义。Welcome和Help界面也可以认为是一种场景，只不过没有用户输入。

Voice应用只是为其他应用程序增加一个入口和控制方式，调用方式还是参照已有的应用。

### 2.2.2系统流程 (画出动态运行流程)

除首次运行的Tutorial界面外，其余界面都可以抽象成Domain（包括Welcome和Help界面），每个Domain流程如下图：



如2 各Domain流程图

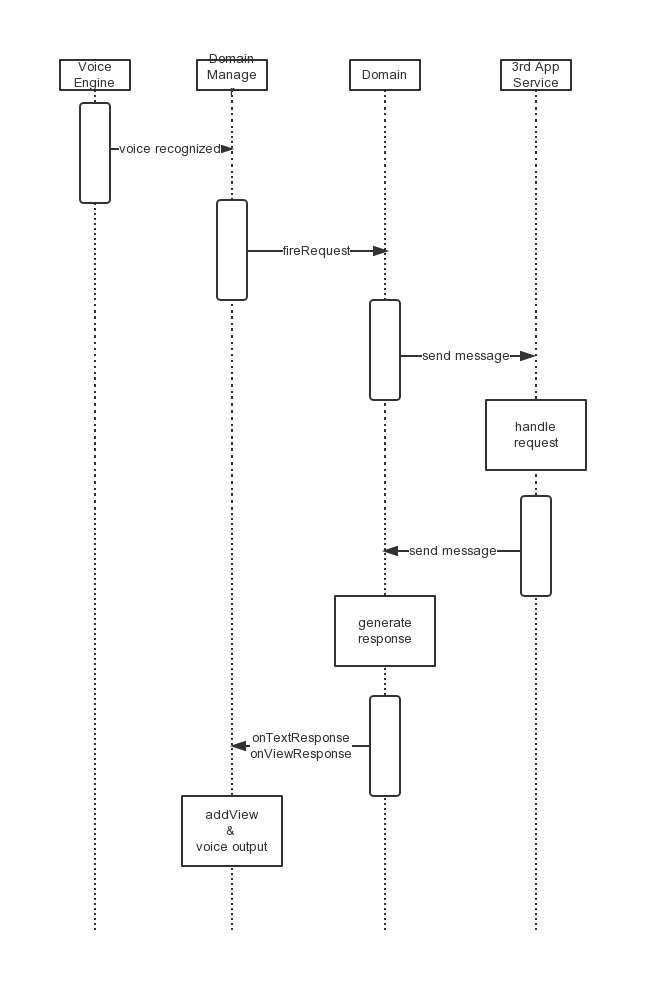


图2 模块时序图

### 2.2.3结构设计

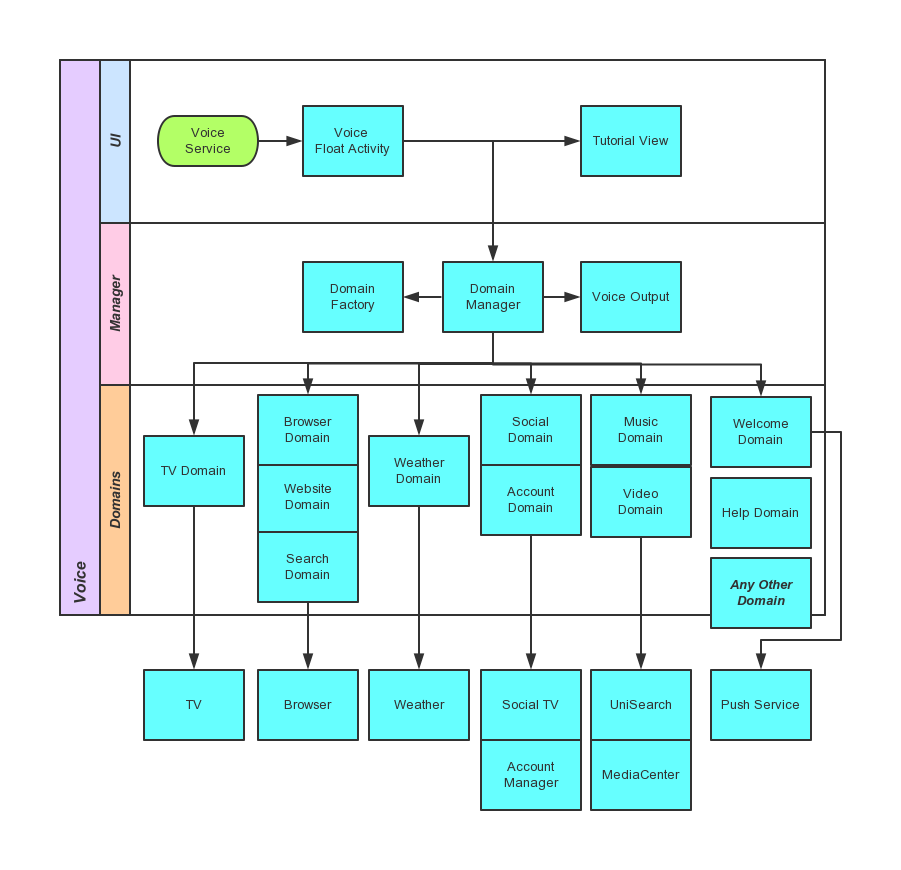


图3 系统结构图

系统整体结构如图3，其中Domain部分只列出目前已知的需要跟其他应用进行交互的Domain。

本系统采用单Service方式，用户请求处理运行在主线程，语音识别引擎和第三方内容运行在各自私有的线程中。

包：com.tpv.xmic.voice.ots2k15

VoiceService： 管理应用生命周期及FloatActivity。

VoiceFloatActivity：显示主界面，实现场景和提示命令轮换、自动隐藏等功能，并预留区域给各场景进行显示；

DomainManager：核心模块，根据各场景的反馈结果组合出界面，加入到浮动窗口，并管理历史界面；

Domain：各场景定义，主要是跟第三方进行通信和渲染各场景的界面，对用户请求进行响应；

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 名称 | 类型 | 功能 |
| 1 | Voice Service | Float Service | 主服务，应用生命期和Float Activity管理 |
| 2 | Voice Float Activity | Float Activity | 引用FWM框架，用于显示界面，包括各Domain界面和Tutorial界面 |
| 3 | Base Domain | Class | 所有Domain的基类 |
| 4 | I Domain Callback | Interface | 各Domain的回调函数接口，返回各场景下的OutTextView或AttachView |
| 5 | Domain Manager | Class | Domain管理器，触发各Domain场景，实现IDomainCallback接口，构造当前场景界面，以及保留历史界面 |
| 6 | Voice Out Tool | Class | Vocalizer语音输出工具类，文本转语音 |
| 7 | Context Utility | Class | 上下文工具类，如判断当前应用和启动应用等 |
| 8 | Domain Factory | Class | 创建Domain对象，工厂模式 |
| 9 | TV Domain | Class | 电视场景，主要是频道和电子节目单功能 |
| 10 | Bowser Domain | Class | 语音浏览器 |
| 11 | Welcome Domain | Class | 每天首次运行的欢迎界面，推送天气或其他信息 |
| 12 | Help Domain | Class | 帮助场景，显示各Domain的典型命令 |
| 13 | Weather Domain | Class | 天气场景，获取指定城市的天气信息 |
| 14 | Social Domain | Class | 社交场景，内容分享功能 |
| 15 | App Domain | Class | 应用场景，应用开启、卸载、搜索等功能 |
| 16 | Website Domain | Class | 网站场景，开启指定网站 |
| 17 | Search Domain | Class | 搜索场景，百度搜索指定关键字 |
| 18 | Instruction Domain | Class | 通用命令，包括各种静态命令 |
| 19 | Music Domain | Class | 音乐场景，搜索在线音乐 |
| 20 | Video Domain | Class | 视频场景，搜索在线视频 |
| 21 | Calendar Domain | Class | 日历场景，显示日历 |
| 22 | Account Domain | Class | 声纹登录场景，打开账户中心 |
| 23 | Error Domain | Class | 全局错误信息场景，如无网络，未识别等 |
| 24 | Tutorial View | View | 向导界面，首次运行时显示 |

表中只列出目前有数据源的Domain，其余Domain由于不确定是否导入暂不列出。

### 2.2.4功能需求与程序的关系 (具体需求与程序模块的对应关系)

|  |  |  |
| --- | --- | --- |
| 序号 | 功能需求(functional spec 和 UI Tree上面体现出来的需求) | 程序组件(对应的实现组件) |
| 1 | 打卡方式（1）- TV (1) | Voice Service / Float Activity |
| 2 | 打卡方式（1）- Tutorial (1.1) | Tutorial View |
| 3 | Start（2）- Start(2) | Welcome Domain |
| 4 | Start（2）- TV/EPG(2.1) | TV Domain |
| 5 | Start（2）- Help(2.2) | Help Domain |
| 6 | Browser(3) –Browser | Voice Service / Float Activity |
| 7 | Browser(3) -Browser （3） | Browser Domain |
| 8 | Function - 天气（4） | Weather Domain |
| 9 | Function - 列车/航班（5） | Train Domain  Flight Domain |
| 10 | Function - 社交（6） | Social Domain |
| 11 | Function - 应用（6） | App Domain |
| 12 | Function - 网站（7） | Website Domain  Search Domain |
| 13 | Function - 通用指令（8） | Instruction Domain |
| 14 | Function - 音乐（9） | Music Domain |
| 15 | Function - 视频（10） | Video Domain |
| 16 | Function - 讲个笑话吧（11） | Joke Domain  Story Domain |
| 17 | Function - 翻译（12） | Translation Domain |
| 18 | Function -人物（13） | Person Domain |
| 19 | Function -文字输入（14） | IME |
| 20 | Function -日历（15） | Calendar Domain |
| 21 | Function -菜谱（16） | Recipe Domain |
| 22 | Function -股票（17） | Stock Domain |
| 23 | Function -影讯（18） | Movie Domain |
| 24 | Function -声纹登陆（19） | Account Domain |
| 25 | Error - Error（1）无网络 | Error Domain |
| 26 | Error - Error（2）未输入/未识别 | Error Domain |

# 3系统数据结构设计

## 3.1数据结构

# 4接口设计

## 4.1用户接口

本系统采用GUI（图形用户界面）与用户交互，用户可以使用鼠标，遥控，键盘与系统交互。不支持命令行交互。

语音控制与手势控制不能同时使用。

## 4.2 外部接口 (与外部程序系统之间交互的接口)

### 4.2.1 语音引擎接口

本系统与语音引擎之间采用广播方式进行通信，广播由引擎方定义，如下：

|  |  |  |  |
| --- | --- | --- | --- |
| **Action** | **Extra** | **Content** | **Description** |
| com.xmic.intent.action.VOICE\_CONTROL\_NLU\_RESULT | text | 识别文本，可能为空 | 识别文本 |
| domainType | command\_domain | 本地命令 |
| error\_domain | 出错 |
| baidu\_domain | baidu类domain（中文）- |
| nuance\_domain | nuance类domain（英文） |
| domain |  | domain名字 |
| jsonStr |  | domain属性及其值，具体根据domain而定，可转为json格式 |

表1 语音引擎发送的广播

### 4.2.2 应用间通信接口

Voice应用与应用间通信方式采用广播或者是Bound Service，总的原则是单向通信采用广播，双向通信采用Bound Service。

应用间通信方式：<http://172.16.144.31:9090/ebony_ui/innovation/flagship_tv_2k15/voice_2k15/doc/不同应用间通信方式调研.docx>

各场景通信参数：<http://172.16.144.31:9090/ebony_ui/innovation/flagship_tv_2k15/voice_2k15/doc/voice各场景数据.xlsx>

通信细节（参数和返回值）：<http://172.16.144.31:9090/ebony_ui/innovation/flagship_tv_2k15/voice_2k15/doc/voice各场景通信细节.docx>

## 4.3 内部接口(内部组件之间调用的接口)

### 4.3.1 Domain接口

每个Domain通过广播或Bound Service方式与第三方应用进行通信，各Domain场景的参数需要综合考虑语音引擎和第三方应用的数据源的能力。

Domain主要接口有：

1. fireRequest，触发请求，一般是向第三方应用发送控制命令或者是获取数据；
2. setCallback：设置回调函数，通知请求处理结果；
3. release：释放Domain场景，例如解绑定服务或取消广播接收器等；

### 4.3.2 DomainManager接口

DomainManager类用于处理语音引擎识别的用户请求和管理历史视图，主要接口有：

1. setContainer：设置容器（ViewGroup，图1红色框部分），每个Domain中界面将加入到这个容器中，最终显示到FloatActivity中；
2. getPrevView：获取上一个历史视图；
3. getNextView：获取下一个历史视图，或者是帮助Domain视图；
4. onTextResponse：请求返回的OutTextView，按场景定义需要转为语音输出；
5. onViewResponse：请求返回的AttachView，显示给用户；

### 4.3.3 ContextUtility接口

ContextUtility静态类是上下文相关的工具类，如判断当前应用、启动应用等，主要接口有：

1. createInstance：创建实例（单例），在主服务中传入上下文；
2. getContext：获取上下文；
3. startActivity：启动指定的应用，指定应用程序的包名和活动类名；
4. getTopActivity：获取当前应用程序；

# 5运行设计

## 5.1 系统初始化（初始化流程图）

## 5.2 运行模块组合

## 5.3 运行时间

# 6出错处理和维护

## 6.1 出错信息

## 6.2 补救措施

## 6.3 系统维护

# 7尚待解决的问题

1. 部分场景缺少内容源，包括：列车、航班、笑话、故事、翻译、人物、菜谱、股票、影讯；
2. 部分场景内容源还未最终确认，包括：社交、音乐、视频；
3. 声纹登录部分依赖账户SDK，具体功能还在讨论中；
4. 社交部分依赖社交SDK，具体功能还在讨论中。