| | |
|---|---|
| 文件标识 | |
| 版 本 号 | V0.1 |
| 密　　级 | |

# IVR 流程开发手册

# 1 目录

# 1 启动 IVRServer

在 ivr 根目录下运行命令

```
./ivrserver&
```

即可后台启动 IVRServer。

# 2 配置文件

## 2.1 系统配置文件 system.conf

```
[general]
ip = 127.0.0.1
port = 9008
reloadport=8813
esllog=0
[conn1]
connstr = 127.0.0.1;test;test
connnum = 10
[freeswitch1]
ip = 127.0.0.1
port = 8021
password = ClueCon
maxsendhandle = 4
[ims1]
ip=127.0.0.1
port=9999
event_port=9121
is_main=1
[calldata]
filename=ivrcalldata
generateinterval=300
generatefilesize=2000
generatepath=ivr/calldata/generate
sendtopath=ivr/calldata/sendto/
```

### general 部分：

- 配置本 IVR 的 IP 地址和端口号，该端口用于与 ivr_loadbalance 通讯；

### connX 部分：

- 配置数据库链接池，需配置连接串和连接池大小。IVR 启动时与 mysql 建立多

条链接保存；

- 可以配置多个连接池，编号不允许重复；

### connX 部分：

- 配置 Freeswitch 的基本信息。IVR 启动时与 Freeswitch 建立多条链接以接收事件；
- 可以配置多个连接池，编号不允许重复；

### calldata 部分：

- 配置 IVR 话单的基本属性；

各个字段的意义如下表所示

| 名称 | 说明 |
|---|---|
| general 部分 | |
| ip | 本地 IP 地址 |
| port | 本地用于通讯的端口 |
| reloadport | 重加载配置的监听端口 |
| Esllog | 是否输出 esl 日志，非 0 输出，0 不输出 |
| connX 部分 | |
| Connstr | 数据库连接字符串用半角的分号分隔，格式为：数据库服务器;用户名;密码 |
| Connnum | 维持的连接数 |
| freeswitchX 部分 | |
| ip | Freeswitch 的 IP 地址 |
| port | Freeswitch 侦听的端口 |
| user | Freeswitch 登录用户名，可省略 |
| password | Freeswitch 登录密码 |
| maxsendhandle | 用于向 Freeswitch 发送消息的链接 |
| Connnum | 维持的连接数 |
| ims 部分 | |
| Ip | Ims 的 ip 地址 |
| Port | Ims 的监听端口 |
| Is_main | 是否是主 ims |
| calldata 部分 | |
| filename | 话单文件前缀 |
| generateinterval | 话单文件生成间隔（单位：秒） |
| generatefilesize | 话单文件生成大小上限（单位：KB） |
| generatepath | 文件生成目录 |
| sendtopath | 文件导出时存储目录 |

## 2.2 呼入流程配置文件 inbound.conf

```
[general]
maxinboundchannum = 10
```

```
[script1]
dnis = 1000
desc =
flowfile = flow/test2_0/testTotal.flow
voxbase =

[-]
desc =
flowfile = flow/inbound1.flow
voxbase =
```

### general 部分：

- 配置呼入流程并发上限；

### scriptX 部分：

- 配置流程，根据被叫号码配置不同流程；
- 流程可配置多个，编号和呼入号码不可重复；

### 默认流程部分([-])：

- 配置默认流程，当呼入号码未匹配任何流程时，启动默认流程
- 默认流程可以为不配置

各个字段的意义如下表所示

| 名称 | 说明 |
|---|---|
| general 部分 | |
| maxinboundchannum | 呼入流程并发上限 |
| scriptX 部分 | |
| dnis | 流程响应的呼入号码 |
| desc | 流程注释信息 |
| flowfile | 流程文件实际存放地址 |
| voxbase | Freeswitch 中 vox 默认根目录 |
| recbase | Freeswitch 中录音根目录，为空则默认为 recbase_ib |
| 默认流程部分 | |
| desc | 流程注释信息 |
| flowfile | 流程文件实际存放地址 |
| voxbase | Freeswitch 中 vox 默认根目录 |
| | |

## 2.3 自启动流程配置文件 autorun.conf

```
[general]
maxautorunchannum= 10

[script1]
```

```
desc =
flowfile = flow/test2_0/testTotal.flow
bill = 1
runchannum = 1
runinterval = 1
```

**general 部分：**

- 配置自启动流程并发上限；

**scriptX 部分：**

- 配置自启动的流程文件；
- 流程可配置多个，编号不可重复；
- 各个字段的意义如下表所示

| 名称 | 说明 |
|---|---|
| general 部分 | |
| maxautorunchannum | 自启动流程并发上限 |
| scriptX 部分 | |
| desc | 流程注释信息 |
| flowfile | 流程文件实际存放地址 |
| bill | 是否写话单，1-是；0-否 |
| runchannum | 单流程同时启动个数 |
| runinterval | 流程每次运行时间间隔（单位：秒） |

## 2.4 日志配置文件 log.conf

配置文件沿用 comlog 配置文件风格，字段意义相同。日志文件共有 4+N 个，其中 4 个为系统默认日志，N 个为自定义日志

- **默认日志：**记录运行 ivr_server 运行情况
  - ivr.log：记录 ivr 运行基本信息，以及各个流程运行信息；
  - esl.log：记录 esl 库收到和发送消息的日志；
  - fsevent.log：记录 IVR 收到 Freeswitch 发送的事件后，转化为 FW 事件的情况；
  - ivrevent.log：记录收到的所有 IVR 事件；
- **自定义日志：**自定义日志等级，用于写日志节点 busiame 的日志。
  - 如自定义 LXB 级别，并定义该级别输出的日志文件，writelog 节点写入的所有 LXB 级别的日志均输出到该文件中。
  - 根据业务需要自定义日志等级。

**所有配置文件的改动，或者流程文件的改动，均需重启 IVRServer 才可以生效。**

# 3 流程简介

呼叫云 IVR 2.0 流程由各种节点构成，节点的不同出口将不同的节点进行串联成有一定

业务逻辑的流程。每个节点的参数可能会用到变量、函数、数据库连接池等辅助信息。一个最简单的业务流程如下：

```
[Node1]
Type=Answer
Callid=${_CALLID}
Descript=摘机
_Success=2
_Failure=0

[Node2]
Type=WaitEvent
Descript=等待应答事件
UUID = VAR_UUID
DATA = VAR_DATA
_Success=2
_Failure=2
_ANSWER=3
_HANGUP=0

[Node3]
Descript=系统挂机
Type=Hangup
Callid=${_CALLID}
_Success=0
_Failure=0
```

**注：**流程文件的格式必须是 unix 文件格式，即：以\n 结尾，而不是\r\n 结尾

## 3.1 节点概述

流程由多个节点构成有向图，每个节点代表一个操作。每个节点包含五个部分，以 Answer 节点为例，各个部分对应关系如下表所示

| 名称 | 对应代码 |
|---|---|
| 声明部分 | [Node1] |
| 类型说明 | type=answer |
| 节点描述 | descript=摘机 |
| 参数部分 | callid=${_CALLID} |
| 出口部分 | _success=2<br>_failure=0 |

所有等号左边的字符均不区分大小写。

- **声明部分**
  - 作用：声明节点标号；

- 注意：
  - 大小写不敏感
  - 开始节点固定为 node1，每个流程里必须由 node1 节点，不能有 node0 节点
  - 若节点重复，以后声明的节点为准

- **类型说明**
  - 作用：说明节点类型，具体见节点部分；
  - 注意：
    - 大小写不敏感
    - 仅可以输入支持的类型，否则 server 无法启动；

- **节点描述**
  - 作用：节点注释，可以输入中文，注释除提醒开发者外，会在 ivr.log 中的 enter 和 leave 日志中打印出来；
  - 注意：
    - 该项可空

- **参数部分**
  - 作用：节点的具体参数设置，参数分为输入参数和输出参数两种类型；
  - 注意：
    - 不同节点的参数个数和意义均不相同，具体请参见各节点说明部分
    - 输入参数均可以使用变量、函数以及组合形式，但需注意输入参数的合法性
    - 输出参数必须为用户指定的变量名

- **出口部分**
  - 作用：指明节点运行完成后的后续执行节点；
  - 注意：
    - 若流程结束，指向 0；
    - 除 0 外，所有的出口指向的节点必须存在，即有向图必须连通；
    - 除等待事件节点外，所有节点只有_success 和_failure 两个出口；

## 3.2 变量

IVR 流程中使用的变量分为系统变量、通道变量、自定义变量三种。引用变量时均使用${变量名}格式。当变量名不存在时，处理为${错误名称}字符串

### 3.2.1 系统变量

每个通道在相同的时间，值一致，变量名称区分大小写。

| 名称 | 含义 | 使用方式 |
|---|---|---|
| _date | 返回当前日期 | ${_date} |

| | | |
|---|---|---|
| _time | 返回当前时间 | ${_time} |
| _second | 返回从 1970-1-1 之后经过的秒数 | ${_second} |
| _inbound_num | 返回当前的呼入通道占用数 | ${_inbound_num} |
| _autorun_num | 返回当前自挨冻通道占用数 | ${_autorun_num} |

### 3.2.2 通道变量

根据每个不同的通道，变量的值不一样，变量名称区分大小写。

| 名称 | 含义 | 使用方式 | 备注 |
|---|---|---|---|
| _ANI | 当前呼叫的主叫 | ${_ANI } | 建议在呼入时保存 |
| _DNIS | 当前呼叫的被叫 | ${_DNIS } | |
| _CALLID | 当前呼叫的 callid | ${_CALLID } | |
| _RECORD_FILE | 当前呼叫的录音文件名称 | ${_RECORD_FILE} | 可以从 record_start 消息中获取 |
| _HANGUP_CAUSE | 当前呼叫挂断原因，可能的原因有：<br>system_hangup：系统挂断<br>user_hangup：用户挂断<br>third_hangup：第三方挂断 | ${_HANGUP_CAUSE} | 可以从挂断消息中区分挂断方 |
| _CALL_BEGIN_TIME | 当前通道被占用时间，格式同 ${_second} | ${_CALL_BEGIN_TIME } | |
| _CALL_ANSWER_TIME | 当前呼叫被应答时间，格式同 ${_second} | ${_CALL_ANSWER_TIME} | |

### 3.2.3 自定义变量

流程开发者可以自定义所需变量，供流程使用。

#### 3.2.3.1 声明

[**var1**]
**type**=string
**name**=srccallid
**initial**=abc

#### 3.2.3.2 说明

| 名称 | 说明 |
|---|---|
| Var1 | 说明下面的定义是一个变量 |
| Type | 变量类型，支持的变量类型包括： |

| | |
|---|---|
| | String：字符串类型<br>Int32：整型<br>Map：映射<br>Resultset：数据集 |
| Name | 变量名称 |
| Initial | 变量初始化名称，仅对 int32 和 string 有效 |
| 注：<br>1） 每个流程内部 var 后面的数字必须保证唯一，否则以后面声明的为准；<br>2） 变量名称在整个流程内部必须保证唯一；<br>3） resultset 变量类型，不能直接赋值（如：rs 仅能被查询语句赋值一次，不允许两个 resultset 类型的变量直接赋值，如：rs1=rs2 这样的方式是不允许的） | |

## 3.3 函数

IVR 提供相应函数供开发者使用。函数的参数可以使用变量，但不允许嵌套。

### 3.3.1 字符串函数

| 名称 | 原型 | 说明 |
|---|---|---|
| LENGTH | String LENGTH(str:string) | 同 string.length() |
| LEFT | String LEFT(str:string,n:uint) | 同 string.substr(0,n) |
| RIGHT | String RIGHT(str:string ,n:uint) | 同 string.substr(LENGTH(str)-n,n) |
| FIND | Int FIND(str:string, str2:string) | 同 string.find(str) |
| SUB | String SUB(str:string, n1:uint, n2:uint)<br>或<br>String SUB(str:string, n1:uint) | 同 string.substr(n1,n2)<br>或<br>string.substr(n1) |
| REPLACE | String SUB(str:string, str1:string, str2:string) | 将字符串 str 中的所有 str1 替换为 str2 |

### 3.3.2 RESULTSET 函数

| 名称 | 原型 | 说明 |
|---|---|---|
| NEXT | Bool next() | Rs.next()返回 bool 变量，标识数据集是否还有记录 |
| VALUE | String value( idx:uint ) | Rs.value(1)，取 rs 当前行的第一列，列标识从 1 开始 |

### 3.3.3 Map 函数

| 名称 | 原型 | 说明 |
|---|---|---|
| VALUE | String VALUE(key :string) | map.vlaue(key)，返回键值为 key 的内容 |

### 3.3.4 其它函数

| 名称 | 原型 | 说明 |
|---|---|---|
| RAND | int RAND(min:uint, max:uint) | 随机数：同 rand()%(max-min+1)+min |

## 3.4 数据库连接池

### 3.4.1 简介

Ivr 流程有需要访问数据库。流程中支持定义连接池，目前支持 mysql 数据库。
声明在配置文件 system.conf 中

### 3.4.2 声明

[**conn**1]
**connstr**=tc-cc-dev01.tc:6666;ccivr;123456
**connnum**=80

### 3.4.3 说明

| 名称 | 说明 |
|---|---|
| conn1 | 说明下面的定义是一个连接池 |
| Connstr | 数据库连接字符串用半角的分号分隔，格式为：数据库服务器;用户名;密码 |
| Connnum | 维持的连接数 |
| 注：①整个流程内部 conn 后面的数字必须保证唯一 | |

# 4 各节点说明

## 4.1 呼叫类

呼叫类节点每次执行后，需要根据需要接“等待事件节点”，用于处理事件。

### 4.1.1 应答（ANSWER）

应答电话。

#### 4.1.1.1 定义

```
[Node1]
Type=Answer
Descript=摘机
Callid=${_CALLID}
_Success=2
_Failure=0
```

#### 4.1.1.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| Callid | 执行指定操作的呼叫 id | Y | |

#### 4.1.1.3 输出参数

无。

#### 4.1.1.4 出口定义

| | |
|---|---|
| _success | 操作成功之后，节点的出口 |
| _failure | 操作失败之后，节点的出口 |

#### 4.1.1.5 注意事项

- 当 callid 字段输入不存在的呼叫 id 时，节点走_failure 出口。
- 应答成功会返回 ANSWER 事件

### 4.1.2 外呼（DIAL）

主动发起外呼。

#### 4.1.2.1 定义

```
[Node2
Type=Dial
Descript=拨号节点
Callid=${_CALLID}
```

**CalledNumber**=1014@127.0.0.1
**CallerNumber**=110
**calltype**=1
**_Success**=2
**_Failure**=0

#### 4.1.2.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| Callid | 执行指定操作的呼叫 id | Y | - |
| Callernumber | 主叫号码 | Y | - |
| Callednumber | 被叫号码 | Y | - |
| Calltype | 呼叫类型：<br>0：呼叫内线<br>非 0：呼叫外线 | Y | - |

#### 4.1.2.3 输出参数

无

#### 4.1.2.4 出口定义

| | |
|---|---|
| _success | 操作成功之后，节点的出口 |
| _failure | 操作失败之后，节点的出口 |

#### 4.1.2.5 注意事项

- 对方振铃时，会返回 ALERT 事件，其中包含振铃的 callid
- 对方接听时，会返回 ANSWER 事件；

### 4.1.3 挂断（HANGUP）

系统挂断某个电话

#### 4.1.3.1 定义

[Node3
**Type**=Hangup

**Descript**=系统挂机
**Callid**=${_CALLID}
**_Success**=0
**_Failure**=0

#### 4.1.3.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| Callid | 执行指定操作的呼叫 id | Y | - |

#### 4.1.3.3 输出参数

无。

#### 4.1.3.4 出口定义

| | |
|---|---|
| _success | 操作成功之后，节点的出口 |
| _failure | 操作失败之后，节点的出口 |

#### 4.1.3.5 注意事项

- 挂断后，会返回 HANGUP 事件和 CHANNEL_DESTROY 事件；
- 两个事件可以根据需要进行定制；

### 4.1.4 桥接（BRIDGE）

将两个电话进行搭接，使双方可以通话。

#### 4.1.4.1 定义

[Node4]
**type**=bridge
**callid1**=${srccallid}
**callid2**=${strcallid}
**_success**=12
**_failure**=13

#### 4.1.4.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| | | | |
| Called1 | 需要桥接的第一个呼叫的 id | Y | - |
| Called2 | 需要桥接的第二个呼叫的 id | Y | - |

#### 4.1.4.3 输出参数

无。

#### 4.1.4.4 出口定义

| | |
|---|---|
| _success | 操作成功之后，节点的出口 |
| _failure | 操作失败之后，节点的出口 |

#### 4.1.4.5 注意事项

- 搭接后，会返回 BRIDGE 事件，表明搭接成功；
- 搭接断开后（如某一方挂断），会返回 UNBRIDGE 事件，目前没有相应节点进行主动断开连接；

### 4.1.5 扩展桥接（BRIDGEEX）

外呼某个号码并直接搭接。

#### 4.1.5.1 定义

```
[Node5
type = Bridgeex
descript = briex
callid = ${inCallid}
callednumber = ${outCallNum}
callernumber = ${gen_showNum}
calltype = 0
bgtype = RING
_success = 53
_failure = 51
```

#### 4.1.5.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| Callid | 执行指定操作的呼叫 id | Y | - |
| callednumber | 外呼的电话号码 | Y | - |
| callernumber | 外呼显示号码 | Y | - |
| calltype | 呼叫类型：<br>0：呼叫内线<br>非 0：呼叫外线 | Y | - |
| bgtype | 等待音类型：<br>RING：语音透传<br>BGM：播放背景音 | Y | - |

#### 4.1.5.3 输出参数

无。

#### 4.1.5.4 出口定义

| | |
|---|---|
| _success | 操作成功之后，节点的出口 |
| _failure | 操作失败之后，节点的出口 |

#### 4.1.5.5 注意事项

- 外呼搭接，会收到 ALERT、ANSWER 等事件

### 4.1.6 申请转座席节点

#### 4.1.6.1 定义

```
[Node1]
Type=RequestAgent
Descript=申请转座席
Callid=${_CALLID}
Timeout=10
ValidTime=5
ReqType=RR_TypeSkill
ReqArgs=咨询
```

**filename**=wait.wav
**_Success**=2
**_Failure**=0

### 4.1.6.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| Callid | 转座席的 callid | Y | |
| Timeout | 请求超时时间 | Y | |
| ValidTime | 设备锁定时间 | Y | |
| ReqType | 请求类型：<br>RR_TypeUnknown,/*未知*/<br>RR_TypeSkill, /*技能*/<br>RR_TypePrivate, /*私人队列*/<br>RR_TypePrivateList, /*私人队列列表*/ | Y | |
| ReqArgs | 请求参数，是工号或者工号列表 | Y | |
| buffer | 业务数据 | N | |
| filename | 排队音文件路径 | N | 默认等待音文件 |

### 4.1.6.3 输出参数

无

### 4.1.6.4 出口定义

| | |
|---|---|
| _success | 操作成功之后，节点的出口 |
| _failure | 操作失败之后，节点的出口 |

### 4.1.6.5 注意事项

无

## 4.1.7 转座席节点

### 4.1.7.1 定义

```
[Node1]
Type=TransAgent
Descript=转座席
TransType= TT_INSTANT
_Success=2
_Failure=0
```

### 4.1.7.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| TransType | 转座席方式<br>TT_INSTANT 立即返回<br>TT_RING 客户振铃后返回<br>TT_ANSWER 客户应答后返回 | Y | |

### 4.1.7.3 输出参数

无

### 4.1.7.4 出口定义

| | |
|---|---|
| _success | 操作成功之后，节点的出口 |
| _failure | 操作失败之后，节点的出口 |

### 4.1.7.5 注意事项

无

## 4.1.8 取消转座席节点

### 4.1.8.1 定义

```
[Node1]
Type= NodeCancelAgent
Descript=取消转座席
_Success=2
_Failure=0
```

### 4.1.8.2 输入参数

### 4.1.8.3 输出参数

### 4.1.8.4 出口定义

| | |
|---|---|
| _success | 操作成功之后，节点的出口 |
| _failure | 操作失败之后，节点的出口 |

### 4.1.8.5 注意事项

## 4.1.9 设置随路数据节点

### 4.1.9.1 定义

```
[Node1]
Type=SetAssociateData
Descript=设置随路数据
key=${data_key}
value=${data_value}
_Success=2
_Failure=0
```

#### 4.1.9.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| key | 随路数据的 key | Y | |
| value | 随路数据的 data | Y | |

#### 4.1.9.3 输出参数

#### 4.1.9.4 出口定义

| | |
|---|---|
| _success | 操作成功之后，节点的出口 |
| _failure | 操作失败之后，节点的出口 |

#### 4.1.9.5 注意事项

### 4.1.10 获取随路数据节点

#### 4.1.10.1 定义

```
[Node1]
Type=GetAssociateData
Descript=获取随路数据
key=${data_key}
value=data_value
_Success=2
_Failure=0
```

#### 4.1.10.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| key | 随路数据的 key | Y | |

#### 4.1.10.3 输出参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| value | 随路数据的 data | Y | |

#### 4.1.10.4 出口定义

| | |
|---|---|
| _success | 操作成功之后，节点的出口 |
| _failure | 操作失败之后，节点的出口 |

#### 4.1.10.5 注意事项

### 4.1.11 申请转座席结果判断节点

#### 4.1.11.1 定义

```
[Node1]
Type=RequestResponseCompare
Descript=判断是否获取了座席
response=${imsres}
agent=agent
_Success=2
_Failure=0
```

#### 4.1.11.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| response | IMS 反馈回的申请转座席结果 | Y | |

#### 4.1.11.3 输出参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| agent | 获取的座席，如果没有获取座席，设置为空字符串””。该字段为字符串 | Y | |

#### 4.1.11.4 出口定义

| | |
|---|---|
| _success | 操作成功之后，节点的出口 |
| _failure | 操作失败之后，节点的出口 |

#### 4.1.11.5 注意事项

## 4.2 计算类

### 4.2.1 赋值（ASSIGN）

将表达式赋值给某个变量

#### 4.2.1.1 定义

```
[node6]
type=assign
exp=${res}.VALUE(number)
result=num
_success=19
_failure=19
```

#### 4.2.1.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| Exp | 原始表达式或原始值 | Y | - |

#### 4.2.1.3 输出参数

| 名称 | 描述 | 是否必需 | 类型 |
|---|---|---|---|
| Result | 被赋值变量名称 | Y | 同 exp |

#### 4.2.1.4 出口定义

| | |
|---|---|
| _success | 操作成功之后，节点的出口 |
| _failure | 操作失败之后，节点的出口 |

#### 4.2.1.5 注意事项

Result 指定的变量，必须在使用之前使用[var*]进行定义。
不要对 resultset 类型的变量进行赋值。

### 4.2.2 比较（COMPARE）

比较两个变量是否相等

#### 4.2.2.1 定义

```
[node7]
type=compare
exp=12345===12345
_success=3
_failure=3
```

#### 4.2.2.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| Exp | 比较表达式，支持的运算符包括：<br>字符串相等：===<br>字符串不等：=!=<br>字符串小于：=<<br>字符串大于：=>><br>字符串小于或等于：=<=<br>字符串大于或等于：=>=<br>数字相等：== | Y | - |

| | | | |
|---|---|---|---|
| | 数字不等：!=<br>数字大于：>=<br>数字小于：<<br>数字大于或等于：>=<br>数字小于或等于：<= | | |

#### 4.2.2.3 输出参数

无。

#### 4.2.2.4 出口定义

| | |
|---|---|
| _success | 操作成功或比较结果为真，节点的出口 |
| _failure | 操作失败或比较结果为假，节点的出口 |

#### 4.2.2.5 注意事项

无。

### 4.2.3 访问数据库（DB）

访问数据库，支持 SQL 语句和存储过程

#### 4.2.3.1 定义

```
[node8]
type=db
connectid=1
sql=call callcloud.p_blacklist('58271288')
#select number from callcloud.blacklist where ani='58271288'
rsvarname=res
_success=2
_failure=0
```

#### 4.2.3.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| connectid | 连接池编号，需在 system.conf 配置文件中配置 | Y | - |

| | | | |
|---|---|---|---|
| Sql | 执行的 sql 语句，支持存储过程 | Y | - |

#### 4.2.3.3 输出参数

| 名称 | 描述 | 是否必需 | 类型 |
|---|---|---|---|
| Rsvarname | 数据集，存放返回数据集对象的变量 | N | resultset |

#### 4.2.3.4 出口定义

| | |
|---|---|
| _success | 操作成功或比较结果为真，节点的出口 |
| _failure | 操作失败或比较结果为假，节点的出口 |

#### 4.2.3.5 注意事项

无。

### 4.2.4 访问 http（HTTP）

发送 http 消息并接收回复

#### 4.2.4.1 定义

```
[node9]
type=http
outtype=POST
url=http://        example.com:8080/
resp=name
_success=20
_failure=20
```

#### 4.2.4.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| outtype | 发送方式：<br>POST：POST 方式，参数直接拼接<br>GET：GET 方式 | Y | - |
| url | 访问的 url 请求串 | Y | - |

#### 4.2.4.3 输出参数

| 名称 | 描述 | 是否必需 | 类型 |
|---|---|---|---|
| Resp | 保存 http 请求的返回值的变量 | Y | string |

#### 4.2.4.4 出口定义

| | |
|---|---|
| _success | 操作成功或比较结果为真，节点的出口 |
| _failure | 操作失败或比较结果为假，节点的出口 |

#### 4.2.4.5 注意事项

- 发送消息的参数均在 url 后面直接拼写，POST 消息不放在消息体中
- 该节点为同步节点，收到消息（5 秒超时）后返回

### 4.2.5 读取 ini 文件（READINI）

读取 ini 配置文件到 map 类型变量中。

#### 4.2.5.1 定义

```
[node10]
type=readini
descript=readini
filename=zhihang/conf/directair.ini
sectionname=${inikey}
mapvarname=ini_val
_success=22
_failure=22
```

#### 4.2.5.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| Filename | Ini 文件 | Y | - |
| Sectionname | 需要读取的 ini 文件的 section 名称 | Y | - |

#### 4.2.5.3 输出参数

| 名称 | 描述 | 是否必需 | 类型 |
|---|---|---|---|
| mapvarname | 保存读取结果的变量 | Y | map |

#### 4.2.5.4 出口定义

| | |
|---|---|
| _success | 操作成功或比较结果为真，节点的出口 |
| _failure | 操作失败或比较结果为假，节点的出口 |

#### 4.2.5.5 注意事项

- 输出参数必须使用 map 类型变量

### 4.2.6 获取 json 串结果（JSONGET）

从 json 串中读取某个字段。

#### 4.2.6.1 定义

```
[node66]
type = JsonGet
descript = 获取返回值
input = ${jsonstr}
key = res
valuetype = BOOL
value = json_true
reason = json_reason
_success = 67
_failure = 68
```

#### 4.2.6.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| input | json 字符串 | Y | - |
| key | 读取的 json 字段名称 | Y | - |
| valuetype | 返回值预计类型：<br>STRING：字符串<br>INT32：数字 | Y | - |

| | | | |
|---|---|---|---|
| | BOOL：bool 值 | | |

#### 4.2.6.3 输出参数

| 名称 | 描述 | 是否必需 | 类型 |
|---|---|---|---|
| value | 保存读取结果的变量 | Y | int32 或 string |
| reason | 结果原因 | Y | int32 |

#### 4.2.6.4 出口定义

| | |
|---|---|
| _success | 操作成功节点的出口 |
| _failure | 操作失败节点的出口 |

#### 4.2.6.5 注意事项

- 原因代码如下：

| 原因码 | 意义 |
|---|---|
| 1 | json 串不合法 |
| 2 | 没有相应的 key 值 |
| 3 | value 值无法转换到对应类型 |
| 4 | 赋值错误 |
| 5 | reson 输出参数不为 INT32 类型 |
| 10 | jsonget array 获取 array 元素时发现为非 array 值 |
| 11 | jsonget array 下标为负数 |
| 12 | jsonget array 中下标值越界 |
| 13 | jsonget array 未找到 key 中下标为%d 的值 |
| 21 | jsonget arraylen 中输出参数不为 INT32 类型 |

### 4.2.7 获取 json 数组长度（JSONGETARRAYLEN）

当 json 串中为 json 数组时，获取数组长度。

#### 4.2.7.1 定义

```
[node66]
type = JsonGetArrayLen
descript = 获取列表长度
Input = ${http_res_body}
key = action
len = call_count_all
```

**reason** = json_reason
**_Success** = 0
**_Failure** = 0

#### 4.2.7.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| input | json 字符串 | Y | - |
| key | 读取的 json 字段名称 | Y | - |

#### 4.2.7.3 输出参数

| 名称 | 描述 | 是否必需 | 类型 |
|---|---|---|---|
| len | 数组总长度 | Y | int32 |
| reason | 结果原因 | Y | int32 |

#### 4.2.7.4 出口定义

| | |
|---|---|
| _success | 操作成功节点的出口 |
| _failure | 操作失败节点的出口 |

#### 4.2.7.5 注意事项

- 当该字段不为 json 数组时，走_failure。
- 原因码见 4.2.6.5 节

### 4.2.8 获取 json 数组结果（JSONGETARRAY）

当 json 串中为 json 数组时，获取数组中指定下表的元素。

#### 4.2.8.1 定义

[node66]
**type** = JsonGetArray
**descript** = 获取返回值
**input** = ${jsonstr}
**key** = res
**index** = 1
**valuetype** = BOOL

**value** = json_true
**reason** = json_reason
**_success** = 67
**_failure** = 68

#### 4.2.8.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| input | json 字符串 | Y | - |
| key | 读取的 json 字段名称 | Y | - |
| index | 读取数组元素下标 | Y | - |
| valuetype | 返回值预计类型：<br>STRING：字符串<br>INT32：数字<br>BOOL：bool 值 | Y | - |

#### 4.2.8.3 输出参数

| 名称 | 描述 | 是否必需 | 类型 |
|---|---|---|---|
| value | 保存读取结果的变量 | Y | int32 或 string |
| reason | 结果原因 | Y | int32 |

#### 4.2.8.4 出口定义

| | |
|---|---|
| _success | 操作成功节点的出口 |
| _failure | 操作失败节点的出口 |

#### 4.2.8.5 注意事项

- 原因码见 4.2.6.5 节

### 4.2.9 获取星期节点

获取当前是周几，输出”1”、”2”、…、”7”。

#### 4.2.9.1 定义

**Type**= WhatWeekday

```
Descript=获取当前周几
weekday=wday
_Success=2
_Failure=0
```

#### 4.2.9.2 输入参数

无

#### 4.2.9.3 输出参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| weekday | 当前时间是周几 | Y | |

周一到周日分别对应”1”、”2”、…、”7”。
输出参数可以是 STRING 或这 INT32 类型。

#### 4.2.9.4 出口定义

| | |
|---|---|
| _success | 操作成功节点的出口 |
| _failure | 操作失败节点的出口 |

#### 4.2.9.5 注意事项

## 4.3 资源媒体

资源媒体类节点每次执行后，需要根据需要接“等待事件节点”，用于处理事件。

### 4.3.1 放音（PLAY）

对某个话路进行放音

#### 4.3.1.1 定义

```
[Node11]
type=play
callid=${_CALLID}
```

**playfile**=hold_music.wav
**allowinterrupt**=1
**_success**=2
**_failure**=2

#### 4.3.1.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| Type | 节点类型，应答为：play | Y | - |
| Callid | 执行指定操作的呼叫 id | Y | - |
| Playfile | 播放的语音文件 | Y | - |
| Allowinterrupt | 是否允许按键打断：<br>0：不允许打断<br>非 0：允许打断 | N | - |

#### 4.3.1.3 输出参数

- 放音结束时返回 RECORD_END 事件；
- 未找到文件返回 FILENOTEXIST 事件

#### 4.3.1.4 出口定义

| | |
|---|---|
| _success | 完成放音之后，节点的出口 |
| _failure | 放音失败，节点的出口 |

#### 4.3.1.5 注意事项

- 放音文件路径为配置文件中的 vox_base+playfile
- 流程中要保证不要向同一个通道连续放音

### 4.3.2 停止放音（STOPMEDIA）

停止某个通话的放音

#### 4.3.2.1 定义

[Node11]
**type**=**stopmedia**
**callid**=${_CALLID}

**_success**=2
**_failure**=2

#### 4.3.2.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| Callid | 停止放音的 callid | Y | - |

#### 4.3.2.3 输出参数

无。

#### 4.3.2.4 出口定义

| | |
|---|---|
| _success | 完成放音之后，节点的出口 |
| _failure | 放音失败，节点的出口 |

#### 4.3.2.5 注意事项

### 4.3.3 播放数字（PLAYNUMBER）

将输入按照数字播放

#### 4.3.3.1 定义

[node90]
**type** = PlayNumber
**descript** = 播放数字
**callid** = ${inCallid}
**playstr** = ${audio_list_len}
**playtype** = int32
**_success** = 91
**_failure** = 91

#### 4.3.3.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| Type | 节点类型，应答为：play | Y | - |
| Callid | 执行指定操作的呼叫 id | Y | - |
| Playstr | 播放的数字串 | Y | - |
| playtype | 播放方式：<br>int32：数字播放（有个十百千万语音，最大支持八位数）<br>number：数字串播放 | Y | - |

#### 4.3.3.3 输出参数

无。

#### 4.3.3.4 出口定义

| | |
|---|---|
| _success | 完成放音之后，节点的出口 |
| _failure | 放音失败，节点的出口 |

#### 4.3.3.5 注意事项

- 播放数字节点默认不允许打断
- 输入的字符串必须全为数字（可以是变量或方法）

### 4.3.4 放音收号（GETDIGITS）

收号节点

#### 4.3.4.1 定义

```
[Node12]
Type=getdigits
Descript=欢迎词
Callid=${_CALLID}
PlayFile=hold_music.wav
maxlen=6
minlen=3
inputtimeout=5
```

**end**=#
**_Success**= 12
**_Failure**= 12

#### 4.3.4.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| Type | 节点类型，应答为：getdigits | Y | - |
| Callid | 执行指定操作的呼叫 id | Y | - |
| playfile | 播放的语音文件 | Y | |
| Manlen | 最大收号长度 | Y | - |
| minlen | 最小收号长度 | Y | - |
| inputtimeout | 两次按键之间的超时 | Y | - |
| end | 未达到最大收号长度的结束按键 | Y | - |

#### 4.3.4.3 输出参数

无

#### 4.3.4.4 出口定义

| | |
|---|---|
| _success | 完成放音之后，节点的出口 |
| _failure | 放音失败，节点的出口 |

#### 4.3.4.5 注意事项

- 放音文件路径为配置文件中的 vox_base+playfile
- 收到的号码通过“GETDIGIT_SUCC 事件”返回；
- 当用户未进行输入超时时，返回“GETDIGIT_FAIL 事件”
- 当用户输入小于“minlen”时，仍然返回“GETDIGIT_SUCC 事件”

### 4.3.5 录音（RECORD）

对某个通话进行录音

#### 4.3.5.1 定义

[Node13]
**Type**=record

**Descript**= record
**callid**=${strcallid}
**recordfile**=${strcallid}.wav
**_Success**=13
**_Failure**=13

#### 4.3.5.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| Callid | 执行指定操作的呼叫 id | Y | - |
| Recordfile | 录音文件 | Y | |

#### 4.3.5.3 输出参数

无。

#### 4.3.5.4 出口定义

| | |
|---|---|
| _success | 完成放音之后，节点的出口 |
| _failure | 放音失败，节点的出口 |

#### 4.3.5.5 注意事项

- 录音默认根目录地址 Freeswitch 中的${sounds_dir}
- 录音开始后返回事件 RECORD_START

### 4.3.6 停止录音（STOPRECORD）

停止某个通话的录音

#### 4.3.6.1 定义

[Node13]
**Type**=**stoprecord**
**Descript**=stoprecord
**callid**=${strcallid}
**_Success**=13
**_Failure**=13

#### 4.3.6.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| Callid | 执行指定操作的呼叫 id | Y | - |

#### 4.3.6.3 输出参数

无。

#### 4.3.6.4 出口定义

| | |
|---|---|
| _success | 完成放音之后，节点的出口 |
| _failure | 放音失败，节点的出口 |

#### 4.3.6.5 注意事项

- 返回 RECORD_END 事件

## 4.4 其它节点

### 4.4.1 注册定时器（REGTIMER）

为当前通道注册一个定时器，定时器到时后返回超时事件

#### 4.4.1.1 定义

```
[node2]
type = RegTimer
descript = 启动定时器，定时器编号为 1,等待 10 秒
timerid = 1
waittime = 10
_success = 0
_failure = 0
```

#### 4.4.1.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|

| | | | |
|---|---|---|---|
| timerid | 计时器标志，仅可以是数字 | Y | - |
| waittime | 超时时间。仅可以是数字，单位为秒 | Y | - |

#### 4.4.1.3 输出参数

#### 4.4.1.4 出口定义

| | |
|---|---|
| _success | 操作成功节点的出口 |
| _failure | 操作失败节点的出口 |

#### 4.4.1.5 注意事项

- 计时器到时时返回 TIMEOUT 事件

### 4.4.2 注销定时器（STOPTIMER）

取消某个定时器

#### 4.4.2.1 定义

```
[node2]
type = StopTimer
descript =停止定时器，定时器编号为 1
timerid = 1
_success = 0
_failure = 0
```

#### 4.4.2.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| timerid | 计时器标志，仅可以是数字 | Y | - |

#### 4.4.2.3 输出参数

#### 4.4.2.4 出口定义

| | |
|---|---|
| _success | 操作成功节点的出口 |

| | |
|---|---|
| _failure | 操作失败节点的出口 |

#### 4.4.2.5 注意事项

无。

### 4.4.3 写日志（WRITELOG）

#### 4.4.3.1 定义

```
[node1]
type = WriteLog
descript = 写日志节点，向“LXB”业务日志中写入“Log Content”
exp = Log Content
businame = LXB
_success = 0
_failure = 0
```

#### 4.4.3.2 输入参数

| 名称 | 描述 | 是否必需 | 默认值 |
|---|---|---|---|
| exp | 写入日志内容 | Y | - |
| businame | 业务名称 | Y | - |

#### 4.4.3.3 输出参数

#### 4.4.3.4 出口定义

| | |
|---|---|
| _success | 操作成功节点的出口 |
| _failure | 操作失败节点的出口 |

#### 4.4.3.5 注意事项

- 业务名称一定要在 log.conf 配置文件中作为自定义日志级别配置好

## 4.4.4 等待事件

等待事件节点，收到事件后会根据事件从不同出口出来

### 4.4.4.1 定义

[node4]
**type** = WaitEvent
**descript** = 等待 IVR 事件
**uuid** = var_uuid
**data** = var_data
**_success** = 0
**_failure** = 0
**_ANSWER** = 0
**_ALERT** = 0
**_HANGUP** = 0
**_PLAY_END** = 0
**_FILENOTEXIST** = 0
**_GETDIGIT_SUCC** = 0
**_GETDIGIT_FAIL** = 0
**_DTMF** = 0
**_RECORD_START** = 0
**_RECORD_END** = 0
**_CHANLE_DESTORY** = 0
**_BRIDGE** = 0
**_UNBRIDGE** = 0
**_TIMEOUT** = 0
**_UNKNOW** = 0
**_IMSRouteResponse=0**

### 4.4.4.2 输入参数

无

### 4.4.4.3 输出参数

| | | | |
|---|---|---|---|
| uuid | 传出内容 1 | Y | - |
| data | 传出内容 2 | Y | - |

#### 4.4.4.4 出口定义

| | |
|---|---|
| _success | 操作成功节点的出口 |
| _failure | 操作失败节点的出口 |

大写字母标志的事件出口可选。

当收到某个事件，但没有相应出口时，从_success 出口出来，此时输出参数无意义。

#### 4.4.4.5 事件对应关系

| 事件名称 | 意义 | uuid | data1 |
|---|---|---|---|
| _ANSWER | 某方应答 | 该事件对应通道的 uuid | 应答的电话号码 |
| _ALERT | 某方振铃 | | 振铃的电话号码 |
| _HANGUP | 某方挂断 | | 挂断的电话号码 |
| _PLAY_END | 播放语音完毕 | | 播放的语音文件路径 |
| _FILENOTEXIST | 未找到播放语音 | | 播放的语音文件路径 |
| _GETDIGIT_SUCC | 收号成功 | | 收到的号码 |
| _GETDIGIT_FAIL | 收号失败 | | 无意义 |
| _DTMF | 收到按键 | | 收到的按键 |
| _RECORD_START | 录音开始 | | 录音文件路径 |
| _RECORD_END | 录音结束 | | 录音文件路径 |
| _CHANLE_DESTORY | Freeswitch 拆线 | | 无意义 |
| _BRIDGE | 搭接成功 | 搭接/拆除的第一个 uuid | 另一通话的 uuid |
| _UNBRIDGE | 拆除成功 | | |
| _TIMEOUT | IVR 超时 | 无意义 | 计时器标志，即 session_timer_id |
| _UNKNOW | 其它未知错误 | | 无意义 |
| _IMSRouteResponse | IMS 转座席反馈 | 该事件对应通道的 uuid | 反馈结果的 json 字符串 |

#### 4.4.4.6 注意事项

- 对于不关注的事件，可以使用_success 出口进行过滤，可以活用该节点。如在搭接后，仅关注 BRIDGE 和 HANGUP 事件，可以如下编写：

```
[node100]
type = WaitEvent
descript =  事件处理-bridge
uuid = evt_uuid
data = evt_data
_success = 100
_failure = 0
```

```
_HANGUP = 101
_BRIDGE = 102
```

将其他所有事件进行过滤，并循环等待事件需要的事件。

**注意：此种写法需要确认一定会得到相应时间，否则会引起死循环**

- 流程编写中，建议均关注 HANGUP 事件。
- 当节点内部处理错误时，走_failure 出口，发生概率很小；
- 每当用户按键时，均会收到 DTMF 事件，使用 getdigits 节点进行收号时，也会收到 DTMF 事件，此时仅关注_GETDIGIT_SUCC、_GETDIGIT_FAIL 即可满足需求