新架构ivr交互查询

1 需求描述

新架构ivr交互查询兼容老架构98,99查询被叫协议，并在此基础上新增了三种协议，分别是95,96,97。

通用的修改需求：

1. 新架构企业，通过ccgeid获取企业信息
2. 新架构企业，确定用户请求和用户响应格式
3. 新架构企业，通过CR-Web提供的查询响应接口返回数据

1.1 95(交互收键模式)

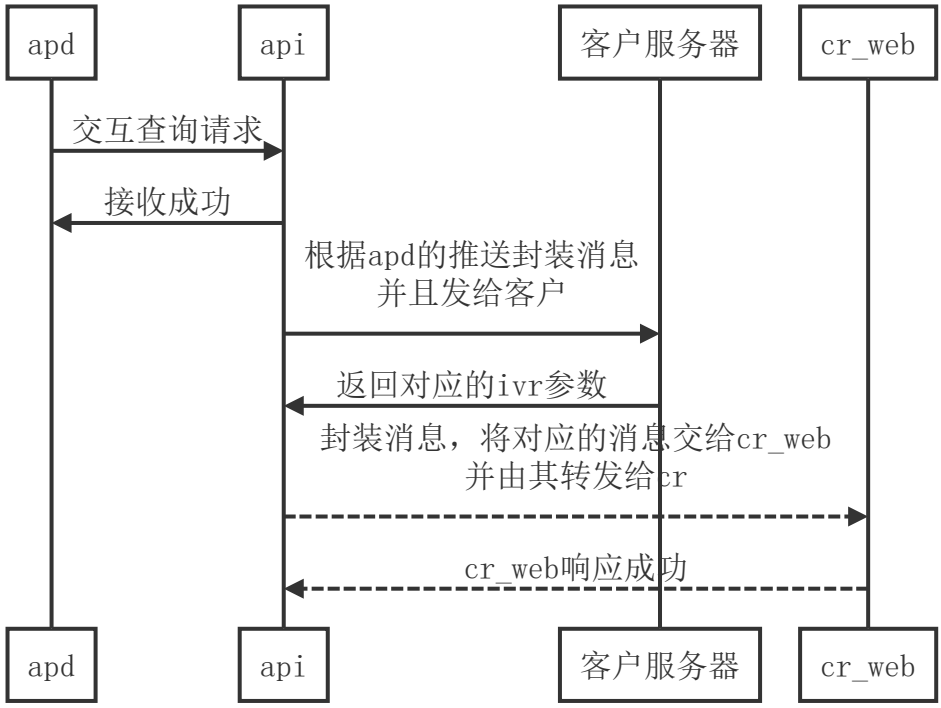
1.2 96(被叫查询模式)

1.3 97(通用交互模式)

1.4 98(AMNB或者AMXNB模式)

1.5 99(AMB或者AMXB模式)

2 交互流程



3 消息定义

3.1 apd->api

3.1.1 95(交互收键模式)

```
{
    "type" : 95,
    "ccgeid" : 1576,
    "callId" : "apixxxxxxxxxxx",
    "ccNumber" : "13260278209conf0_1519439781636",
    "callType" : 0,
    "ivrFlowId" : 245,
    "ivrQueryId" : 1,
    "caller" : "13260278209",
    "callerType" : 1,
    "switchNumber" : "02566699794",
    "called" : "1001",
    "calledType" : 2,
    "timestamp" : "1519439787",
    "userQueryId" : "id_0000001",
    "inputKeys" : "1000",
    "variables" : [
        {"id_number" : "110108198703127621" },
        {"name" :"" },
        {"address":"" }
    ]
}

```

3.1.2 96(被叫查询模式)

```
{
    "type" : 96,
    "ccgeid" : 1576,
    "callId" : "apixxxxxxxxxxx",
    "ccNumber" : "13260278209conf0_1519439781636",
    "callType" : 0,
    "ivrFlowId" : 245,
    "ivrQueryId" : 1,
    "caller" : "13260278209",
    "callerType" : 1,
    "switchNumber" : "02566699794",
    "called" : "1001",
    "calledType" : 2,
    "timestamp" : "1519439787",
    "userQueryId" : "id_0000001",
    "inputKeys" : "1000",
    "variables" : [
        {"id_number" : "110108198703127621" },
        {"name": "" },
        {"address": "" }
    ]
}
```

3.1.3 97(通用交互模式)

```

```

```
{
    "type" : 97,
    "ccgeid" : 1576,
    "callId" : "apixxxxxxxxxxx",
    "ccNumber" : "13260278209conf0_1519439781636",
    "callType" : 0,
    "ivrFlowId" : 245,
    "ivrQueryId" : 1,
    "caller" : "13260278209",
    "callerType" : 1,
    "switchNumber" : "02566699794",
    "called" : "1001",
    "calledType" : 2,
    "timestamp" : "1519439787",
    "userQueryId" : "id_0000001",
    "inputKeys" : "1000",
    "variables" : [
        {"id_number" : "110108198703127621" },
        {"name" :"" },
        {"address":"" }
    ]
}
```

3.1.4 98(AMNB或者AMXNB模式)

```
{
    "type" : 98,
    "ccgeid" : 1576,
    "callId" : "apixxxxxxxxxxx",
    "ccNumber" : "13260278209conf0_1519439781636",
    "ivrFlowId" : 34,
    "ivrQueryId" : 35,
    "switchNumber" : "02566699794",
    "useNumber" : "02566699794",
    "caller" : "13260278209",
    "callerType" : 1,
    "timestamp" : "1519439787",
    "path" : "1000"
}
```

3.1.5 99(AMB或者AMXB模式)

```
{
    "type" : 99,
    "ccgeid" : 1576,
    "callId" : "apixxxxxxxxxxx",
    "ccNumber" : "13260278209conf0_1519439781636",
    "ivrFlowId" : 34,
    "ivrQueryId" : 35,
    "switchNumber" : "02566699794",
    "useNumber" : "02566699794",
    "caller" : "13260278209",
    "callerType" : 1,
    "timestamp" : "1519439787",
    "path" : "1000"
}
```

3.2 api内部消息转换

根据前面的需求描述

1. 根据ccgeid从企业表中获取企业信息

根据ccgeid获取企业信息，我们不需要通过callId或者总机号才能获取到企业信息，需要注意的是。后面处理时还会尝试获取一次企业的详细信息，对于新架构，我们需要将企业的信息发送过去，以减少一次数据库操作（但是enterprise结构体的大小对于推送的消息队列来说过于庞大了）。

2. 新架构的通用ivr交互新增了部分通用字段，有些字段api必须转发给客户。需要新增一个结构体用来存储这些信息

通话类型	IVR流程	callType	caller	callerType	called	calledType
呼入	正常流程	5	客户号码	1	-	-
呼入	子IVR流程	5	客户号码	1	坐席话机号	1/2
呼出	子IVR流程	0或7	坐席话机号	1/2	客户号码	1
语音通知	语音通知流程	3	总机号码	1	客户号码	1

```
typedef struct __push_post_data_ivr_moudle__
{
    DB_CALL_RECORD_CALL_TYPE    calltype;     //联合calltype caller called，可以得知在此次ivr请求中，哪个是客户，哪个是坐席
    /*
    | 通话类型 | IVR流程       | callType | caller     | callerType | called     | calledType |
    | -------- | ------------ | -------- | ---------- | ---------- | ---------- | ---------- |
    | 呼入     | 正常流程      | 5        | 客户号码    | 1          | -          | -          |
    | 呼入     | 子IVR流程     | 5        | 客户号码    | 1          | 坐席话机号  | 1/2        |
    | 呼出     | 子IVR流程     | 0或7     | 坐席话机号  | 1/2        | 客户号码    | 1          |
    | 语音通知 | 语音通知流程  | 3        | 总机号码    | 1          | 客户号码    | 1          |
    */
    unsigned long ivrFlowId;    //ivr流程id
    unsigned long ivrQueryId;   //ivr查询节点id
    char          userQueryId[64];  //用于向客户服务器确定id对应下一步的操作是什么
    char          inputKeys[64];    //推送上次输入的按键
    char          *variables;
    /*
    用户自定义查询请求变量值集合，是{"variables" : [ {"id_number" : "110108198703127621"}, {:"name" :""}, {"address":""} ]},
    */
```

```
}ModCallPushPostDataIVRMoudle;

typedef struct __push_post_data__
{
    unsigned char               callId[CALL_RECORD_CALLID_MAX_LEN];
    unsigned char               ccNumber[CALL_RECORD_CALLID_MAX_LEN];
    unsigned char               caller[PHONE_NUMBER_MAX_LEN];
    unsigned char               called[PHONE_NUMBER_MAX_LEN];
    int                         xferTimes;
    BOOL                        extCaller;
    BOOL                        extCalled;
    BOOL                        isCaller;
    DB_CALL_RECORD_CALL_TYPE    type;
    DB_CALL_RECORD_STATUS       status;
    unsigned char               useNumber[PHONE_NUMBER_MAX_LEN];
    unsigned char               switchNumber[PHONE_NUMBER_MAX_LEN];
    unsigned char               subNumber[PHONE_NUMBER_MAX_LEN];
    unsigned char               virtNumber[PHONE_NUMBER_MAX_LEN];
    unsigned long               enterpriseId;
    unsigned long               ccgeid;
    unsigned long               ringTime;
    unsigned long               startTime;
    unsigned long               endTime;
    unsigned long               timestamp;
    unsigned long               duration;
    unsigned long               reason;
    unsigned long               gid;
    unsigned long               pbxCallLogId;
    char                        feedback[CALL_RECORD_FEEDBACK_MAX_LEN];
    ModCallPushPostDataIVRMoudle    *ivr_argv;      //新架构通用ivr扩展字段

    BOOL                        realtimeData;
    BOOL                        isCheckData;
    BOOL                        isCommonEnterprise;
    BOOL                        hangup2calling;
    int                         index;
    int                         lwpid;
    int                         mes_type;
    int                         failed_delay_time;
    unsigned char               path[CALL_RECORD_FEEDBACK_MAX_LEN];   //99协议 ，按键 (可能包含二级按键，比如:2-9)

    unsigned long               app_id;
    unsigned long               provinceId;
    char                        number[USER_NUMBER_MAX_LEN];
    char                        mobile[PHONE_NUMBER_MAX_LEN];
    char                        destNumber[USER_NUMBER_MAX_LEN];
    unsigned long               ngnReason;
 //201708-N02细化通话失败和挂断原因
    char                        batchCallId[CALL_RECORD_CALLID_MAX_LEN];
    char                        batchCallUserData[USER_DATA_MAX_LEN];
    char                        batchCallTaskId[BATCH_TASK_ID_MAX_LEN];
} ModCallPushPostData;

typedef struct __app_server_callback_argument__
{
    BOOL    isPiccCallbackArg;
    BOOL    isCommonEnterprise;
```

```
    BOOL    packageBalanceEnough;
    int     callbackTimes;
    int     appCallbackDataFormat;
    int     call_detail_num;
    unsigned long app_id;
    unsigned long provinceId;
    unsigned long enterpriseId;

    unsigned long curTimestamp;
    unsigned long timestamp;
    unsigned char keyFeedback[CALL_RECORD_FEEDBACK_MAX_LEN];
    unsigned char destNumber[USER_NUMBER_MAX_LEN];
    PiccCallBackArgument *piccArgs;

    ModCallPushPostDataIVRMoudle    *ivr_argv;      //新架构通用ivr扩展字段,记得释放
    DB_CALL_RECORD_CALL_TYPE    type;       //此次推送的通话类型，根据ivr文档描述call_records和call_detail中的type已经不能标识这次的推送通话类型了

    APP_CALLBACK_TYPE   callbackType;
    char AppServerUrl[CALLBACK_URL_MAX_LEN];
    // MYSQL *pDb;                                  // Only for local database access

    // ModCallPushPostData *post_data;
    EMICALLDEV_DB_CALL_RECORDS  *call_record;
    EMICALLDEV_DB_CALL_DETAILS  *call_details;
    EMICALLDEV_DB_CALL_DETAILS  *cur_call_detail;
    EMICALLDEV_DB_ACCOUNTS      *mainAccount;
    EMICALLDEV_DB_APPLICATIONS  *appInfo;
    EMICDEV_DB_COMMON_ENTERPRISES   *comm_enterprise;
    EMICDEV_DB_ENTERPRISES      *enterprise;
    ENTERPRISE_BASE_INFO        *eInfo;
    EMICALLDEV_DB_CALL_RECORD_CC_EXTENDS  *call_record_extends;

    char downloadUrl[HTTP_URL_MAX_LEN];
} AppServerCallbackArgument;
```

3.2 api->用户服务器

请求格式和用户响应格式由客户进行配置，可以是json或者xml

3.2.1 请求客户

1. 99/98 99和98向客户的请求信息保持不变

json

```
{
    'appId':'b23abb6d451346efa13370172d1921ef',
    'callId':'api1234059445aDbbJxIdbT',
    'accountSid':'c5dc4b87f33ef2ef37c8e974793ad8e5',
    'caller':'18769874345',
    'path':'1-2',
    'type':0,
    'callType':99,
    'useNumber':'02566687987',
    'switchNumber' : "02566699794",
    'userData':"FE87D3"
}
```

2. 95,96和97消息需要使用新架构新的消息格式

新架构的查询场景不再限于呼入场景，api客户可能需要知道主叫或被叫是坐席还是客户，callType需要传给客户

```
{
    "type" : 95/96/97,
    //"initialCallType": 1,
    //一开始发起的通话类型
    "callType" : 1,
    "accountSid": "c5dc4b87f33ef2ef37c8e974793ad8e5",
    "subAccountSid": "c5dc4b87f33ef2ef37c8e974793ad8e5",
    'appId':'b23abb6d451346efa13370172d1921ef',

    "callId" : "apixxxxxxxxxxx",
    "caller" : "13260278209",
    "called" : "1001",
    "useNumber" : "02566699794",
    "userQueryId":"id_0000002",
    "inputKeys" : "1000",
    "variables" : [
        {"id_number" : "110108198703127621" },
        {"name" :"" },
        {"address":"" }
    ],
    'userData':"FE87D3"

}
```

说明：在上例中，有3个全局变量：id_number、name和address，id_number已经赋值，name和address未赋值，需要用户服务器返回，并在IVR其它节点中引用。

```
<request>
    <callType>5</callType>
    <type>97</type>
    <accountSid>42f7de84ff2ea9b4d71c2aa667455249</accountSid>
    <subAccountSid>51d17d870e0c1da85869580195b31f1d</subAccountSid>
    <appId>68a87b1250011b35cfb244f04a27ca9d</appId>
    <callId>1582511927.526072</callId><caller>15861800293</caller>
    <called>0252133</called>
    <useNumber>02566687671</useNumber><usrQueryId>1</usrQueryId>
    <inputKeys>1</inputKeys>
    <userData></userData>
    <variables>
```

```
        <name1>value1</name1>
        <name2>value2</name2>
    </variables>
</request>
```

3.2.2 客户响应

3.2.2.1 98/99返回数据

```
{
    "response":{
        //0:允许通话，否则失败
        "retcode": 0,
        //返回被叫
        "action": 0,
        "called": "1****211",
        "number": "1002",
        "workNumber": "test",
        "waitTime": 20,
        "outNumber":"02566687**1",
        "reason": "原因描述",
        "userdata": "用户数据"
    }
}
//其中number>workNumber>called,这三个都是被叫选项，三者选一。被叫如果传多个，会按顺序拨打直至拨打完毕
```

3.2.2.2 95/96/97

通用格式

```
{
    "response":{
        "retcode": 0,
        //客户响应数据都将透传给cr，具体含义见api->cr_web的定义
        //可以为空，也可以和请求保重的userQueryId相同，也可以不同
        //不为空时，下次的交互查询节点会直接赋值userQueryId
        "userQueryId":"id_0000002",

        //variables是请求包中variables的子集，需要赋值的参数必须要通过这个列表中返回
        "variables": [
            {"id_number" : "110108198703127621" },
            {"name" :"" },
            {"address":"" }
        ],
        //虚拟键值，交互收键模式时的响应参数
        "virtualKey":"1111",
        //用户返回的下一步参数，这个数据应当由api透传给cr，现cr已经定义好响应数据，所以和需求文档中的格式略有不同，具体数据参数见api->cr_web的消息定义
        "nextAction" : {
            "action" : 1,
            "params" : {
                "voiceId" : "播放语音文件id",
                "voiceName" : "播放语音文件唯一名称",
                "allowBreak" : "是否允许打断: 0-不允许 1-允许"
```

```
            }
        },
        "userData":"FE87D3"
    }
}
```

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <retcode>0</retcode>
    <userQueryId>1</userQueryId>
    <userData>FE87D3</userData>
    <variables>
        <a725>1582726762</a725>s
        <as725>1582726762</as725>
    </variables>

    <nextAction>
        <action>2</action>
        <params>
            <voiceId>1</voiceId>
            <voiceName>dwad</voiceName>
            <allowBreak>1</allowBreak>
        </params>
    </nextAction>
</response>
```

3.2.2.2.1 nextAction放音

json

```
"nextAction" : {
    "action" : 1,
    "params" : {
        //下面四个参数选择其中之一
        "voiceId" : "播放语音文件id",
        "voiceName" : "播放语音文件唯一名称",
        "voiceTempId": "播放语音模板id",
        "voiceTempName": "播放语音模板名称",
        //语音模板参数，多个参数见用英文","号隔开，当选择voiceTempId或者voiceTempName时需要传入
        "voiceTempParams": "语音模板参数",

        "allowBreak" : "是否允许打断: 0-不允许 1-允许"
    }
}
```

xml

```xml
<nextAction>
    <action>2</action>
    <params>
        <voiceId>1</voiceId>
        <voiceName>dwad</voiceName>
        <allowBreak>1</allowBreak>
    </params>
</nextAction>
```

3.2.2.2.2 nextAction放音收键

json

```json
"nextAction" : {
    "action" : 2,
    "params" : {
        //下面四个参数选择其中之一
        "voiceId" : "播放语音文件id",
        "voiceName" : "播放语音文件唯一名称",
        "voiceTempId": "播放语音模板id",
        "voiceTempName": "播放语音模板名称",
        //语音模板参数，多个参数见用英文","号隔开，当选择voiceTempId或者voiceTempName时需要传入
        "voiceTempParams": "语音模板参数",

        "allowBreak" : "是否允许打断: 0-不允许 1-允许",
        "getKeyNumber" : "获取按键位数",
        "getKeyTimeout" : "收键超时时间",
        "endWithHashKey" : "是否以#号键结束, 0-不是, 1-是"
    }
}
```

3.2.2.2.3 nextAction转技能组

```json
"nextAction" : {
    "action" : 3,
    "params" : {
        "acdId" : "技能组id",
        "acdName" : "技能组名称",
        "useAcdValue" : "0-不使用技能组配置 1-使用技能组配置",
        "queueTime" : "排队超时时长",
        "switchTimes" : "坐席流转次数",
        "ringTimeout" : "坐席振铃超时时长",
        "customerMemory" : "0-不记忆 1-优先熟客记忆 2-强制熟客记忆"
    }
}
```

3.2.2.2.4 nextAction转坐席

```
"nextAction" : {
    "action" : 4,
    "params" : {
        "workNumber" : "1001,1002,1003",
        "number" : "1001,1002,1003",
        "queueTime" : "坐席忙时排队时长",
        "ringTimeout" : "多坐席情况下，坐席振铃超时时长",
    }
}
```

3.2.2.2.5 nextAction转外线

```
"nextAction" : {
    "action" : 5,
    "params" : {
        "called" : "外线被叫号码",
        "outNumber" : "呼出总机号码"
    }
}
```

3.2.2.2.6 nextAction转其他IVR流程

```
"nextAction" : {
    "action" : 6,
    "params" : {
        "ivrFlowId" : "IVR流程id",
        "ivrFlowName" : "IVR流程名称"
    }
}
```

3.2.2.2.7 nextAction结束IVR流程

```
"nextAction" : {
    "action" : 7
}
```

3.3 api->cr_web

新架构cr采用统一的json格式，所以不管是95、96、97、98还是99，api需要将客户返回的消息转换成cr需要的格式，而不需要校验。api需要在对外文档中列出这些响应数据对应的格式。

以下是cr根据action定义的响应消息内容，需要根据这些重新生成给cr的json响应，然后通过cr_web透传给cr

为方便路由，要求将ccgeid和cc_number作为头域参数

action

```
typedef enum
{
    0,  invalid
    1,  放音响应
    2,  放音按键响应
    3,  转技能组响应
    4,  转坐席响应
    5,  转外线响应
    6,  转其他IVR流程响应
    7,  流程结束响应
}IvrActionType;
```

3.3.1 交互收键模式响应消息

```
{
    "eid" : "00011",
    "ccgeid" : "111222",
    "request_type" : "1",    //标识是通话控制请求还是交互式ivr响应
    "class_type" : "1",      //
    "ccNumber" : "",         //
    "data" :
    "
    {
        "rspCode" : 0,
        "userQueryId":"id_0000002",
        "virtualKey" : "5",
        "variables" : [
             { "id_number" : "110108198703127621" },
             { "name" :"张三" },
             { "address":"江苏省南京市江宁区" }
        ]
        "reason" : "test",
        "userdata" : "test"
    }
    "
}
```

3.3.2 被叫查询和通用交互模式响应消息

3.3.2.1 放音响应

json to cr

```
{
    "eid" : "00011",
    "ccgeid" : "111222",
    "request_type" : "1",    //标识是通话控制请求还是交互式ivr响应
    "class_type" : "1",      //
    "ccNumber" : "",         //
    "data":"
        "rspCode" : 0,
        "userQueryId":"id_0000002",
        "variables" : [
            { "id_number" : "110108198703127621" },
```

```
            { "name" :"张三" },
            { "address":"江苏省南京市江宁区" }
        ],
        "nextAction" : {
            "action" : 1,
            "params" : {
                //下面四个参数选择其中之一
                "voiceId" : "播放语音文件id",
                "voiceName" : "播放语音文件唯一名称",
                "voiceTempId": "播放语音模板id",
                "voiceTempName": "播放语音模板名称",
                //语音模板参数，多个参数见用英文","号隔开，当选择voiceTempId或者
voiceTempName时需要传入
                "voiceTempParams": "语音模板参数",

                "allowBreak" : "是否允许打断: 0-不允许 1-允许"
            }
        },
        "reason" : "test",
        "userdata" : "test"
    "
}
```

3.3.2.2 放音收键响应

json to cr

```
{
    "eid" : "00011",
    "ccgeid" : "111222",
    "request_type" : "1",    //标识是通话控制请求还是交互式ivr响应
    "class_type" : "1",      //
    "ccNumber" : "",         //
    "data":"
        "rspCode" : 0,
        "userQueryId":"id_0000002",
        "variables" : [
            { "id_number" : "110108198703127621" },
            { "name" :"张三" },
            { "address":"江苏省南京市江宁区" }
        ],
        "nextAction" : {
            "action" : 2,
            "params" : {
                //下面四个参数选择其中之一
                "voiceId" : "播放语音文件id",
                "voiceName" : "播放语音文件唯一名称",
                "voiceTempId": "播放语音模板id",
                "voiceTempName": "播放语音模板名称",
                //语音模板参数，多个参数见用英文","号隔开，当选择voiceTempId或者
voiceTempName时需要传入
                "voiceTempParams": "语音模板参数",

                "allowBreak" : "是否允许打断: 0-不允许 1-允许",
                "getKeyNumber" : "获取按键位数",
                "getKeyTimeout" : "收键超时时间",
                "endWithHashKey" : "是否以#号键结束, 0-不是, 1-是"
```

```
            }
        },
        "reason" : "test",
        "userdata" : "test"
}
```

3.3.2.3 转技能组响应

json to cr

```
{
    "eid" : "00011",
    "ccgeid" : "111222",
    "request_type" : "1",    //标识是通话控制请求还是交互式ivr响应
    "class_type" : "1",      //
    "ccNumber" : "",         //
    "data":"
        "rspCode" : 0,
        "userQueryId":"id_0000002",
        "variables" : [
            { "id_number" : "110108198703127621" },
            { "name" :"张三" },
            { "address":"江苏省南京市江宁区" }
        ]
        "nextAction" : {
            "action" : 3,
            "params" : {
                "acdId" : "技能组id",
                "acdName" : "技能组名称",
                "useAcdValue" : "0-不使用技能组配置 1-使用技能组配置",
                "queueTime" : "排队超时时长",
                "switchTimes" : "坐席流转次数",
                "ringTimeout" : "坐席振铃超时时长",
                "customerMemory" : "0-不记忆 1-优先熟客记忆 2-强制熟客记忆"
            }
        }
        "reason" : "test",
        "userdata" : "test"
    "
}
```

3.3.2.4 转座席响应

json to cr

```
{
    "eid" : "00011",
    "ccgeid" : "111222",
    "request_type" : "1",    //标识是通话控制请求还是交互式ivr响应
    "class_type" : "1",      //
    "ccNumber" : "",         //
    "data":"
        "rspCode" : 0,
        "userQueryId":"id_0000002",
        "variables" : [
            { "id_number" : "110108198703127621" },
            { "name" :"张三" },
```

```
            { "address":"江苏省南京市江宁区" }
        ]
        "nextAction" : {
            "action" : 4,
            "params" : {
                "workNumber" : "1001,1002,1003",
                "number" : "1001,1002,1003",
                "queueTime" : "坐席忙时排队时长",
                "ringTimeout" : "多坐席情况下，坐席振铃超时时长"
            }
        },
        "reason" : "test",
        "userdata" : "test"
    "
}
```

3.3.2.5 转外线响应

json to cr

```
{
    "eid" : "00011",
    "ccgeid" : "111222",
    "request_type" : "1",    //标识是通话控制请求还是交互式ivr响应
    "class_type" : "1",      //
    "ccNumber" : "",         //
    "data":"
        "rspCode" : 0,
        "userQueryId":"id_0000002",
        "variables" : [
            { "id_number" : "110108198703127621" },
            { "name" :"张三" },
            { "address":"江苏省南京市江宁区" }
        ]
        "nextAction" : {
            "action" : 5,
            "params" : {
                "called" : "外线被叫号码",
                "outNumber" : "呼出总机号码"
            }
        }
        "reason" : "test",
        "userdata" : "test"
    "
}
```

3.3.2.6 转其他IVR流程响应

json to cr

```
{
    "eid" : "00011",
    "ccgeid" : "111222",
    "request_type" : "1",    //标识是通话控制请求还是交互式ivr响应
    "class_type" : "1",      //
    "ccNumber" : "",         //
    "data":"
```

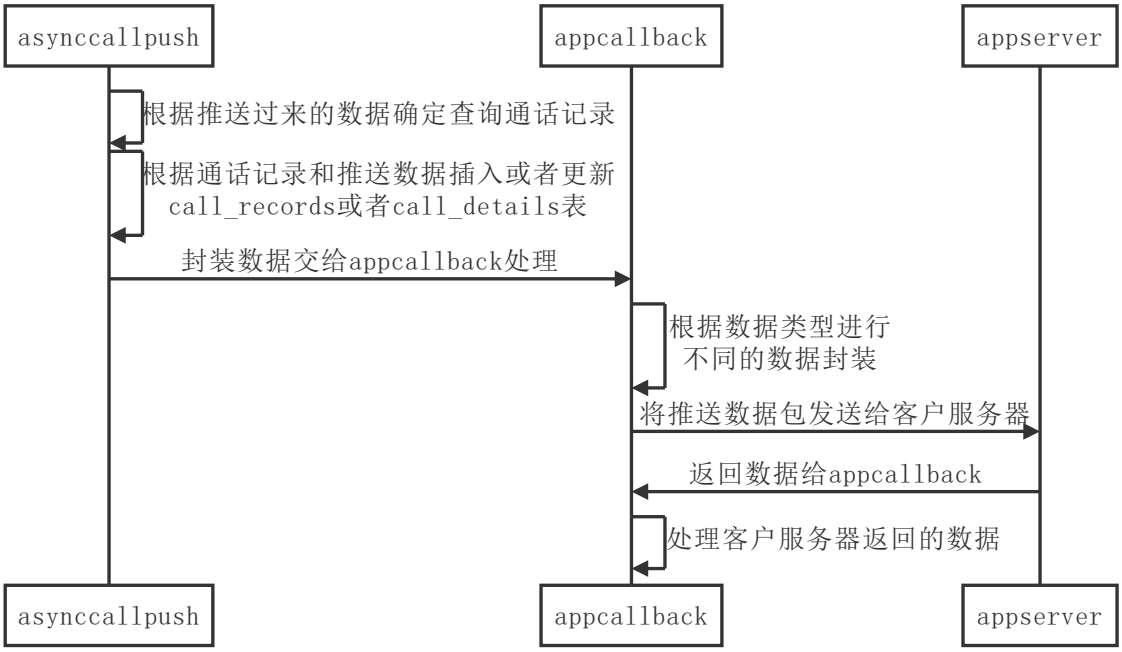
```
        "rspCode" : 0,
        "userQueryId":"id_0000002",
        "variables" : [
            { "id_number" : "110108198703127621" },
            { "name" :"张三" },
            { "address":"江苏省南京市江宁区" }
        ]
        "nextAction" : {
            "action" : 6,
            "params" : {
                "ivrFlowId" : "IVR流程id",
                "ivrFlowName" : "IVR流程名称"
            }
        }
        "reason" : "test",
        "userdata" : "test"
    "
}
```

3.3.2.7 流程结束响应

```
{
    "eid" : "00011",
    "ccgeid" : "111222",
    "request_type" : "1",    //标识是通话控制请求还是交互式ivr响应
    "class_type" : "1",      //
    "ccNumber" : "",         //
    "data":"
        "rspCode" : 0,
        "userQueryId":"id_0000002",
        "variables" : [
            { "id_number" : "110108198703127621" },
            { "name" :"张三" },
            { "address":"江苏省南京市江宁区" }
        ]
        "nextAction" : {
            "action" : 7
        }
        "reason" : "test",
        "userdata" : "test"
    "
}
```

4 处理流程

- api callpush模块时序图



asynccallpush到appcallback只需要将ivr的特有数据透传就行。

appcallback到appserver因为type和calltype的含义已经变化，在原有的结构上进行扩展已经不适合了。下面是appcallback的新逻辑。

```
//
```

经过考察我们主要的修改是：

1. 发向appcallback队列的数据会进行扩展，存储ivr扩展信息。
2. 对于交互类型的数据，不能更新通话记录
3. 回调接口扩展，新增回调函数
4. 查询结果封装并返回，通过cr_web接口

疑问：

1. 98和99或者96和以前不同，会推送振铃请求
2. 目前type和calltype的定义比较混乱
3. 不支持重传

4.1 95(交互收键模式)

如果已有通话记录，则不对通话记录进行任何修改；如果没有通话记录，则该通话一定是呼入通话，需要在call_records和call_details表中生成原始通话记录，通话类型等于请求数据中的callType

4.2 96(被叫查询模式)

被叫查询api会直接根据消息数据入库，和之前的处理方式保持一致即可。

4.3 97(通用交互模式)

如果已有通话记录，则不对通话记录进行任何修改；如果没有通话记录，则该通话一定是呼入通话，需要在call_records和call_details表中生成原始通话记录，通话类型等于请求数据中的callType

4.4 98(AMNB或者AMXNB模式)

被叫查询api会直接根据消息数据入库，和之前的处理方式保持一致即可。

4.5 99(AMB或者AMXB模式)

被叫查询api会直接根据消息数据入库，和之前的处理方式保持一致即可。

5 涉及代码