概述

易米电话工具条api接口文件把易米工具条文件剥离了UI相关部分代码，方便用户直接通过程序调用易米工具条服务。

使用介绍

我们提供两种方式使用api接口，并提供相应示例代码。

1. 传统方式，在HTML页面中通过 `script` 标签引入api接口文件

   ```
   <script src="https://unpkg.com/jssip-emicnet/dist/phonebar.js"></script>
   <script>
    //调用 api 接口，可参见示例
   </script>
   ```

   但由于工具条api代码较复杂，我们建议用下面这种方式：

2. npm 安装易米工具条，通过各种打包编译工具使用, webpack 或gulp使用

   ```
   1. //安装易米工具条包
   npm i jssip-emicnet
   2. 代码中引入
   let phonebar = require("jssip-emicnet/dist/phonebar");
   //或者
   import phonebar from 'jssip-emicnet/dist/phonebar';
   3. 注意 babel 设置，比如 target，比如"last 5 Chrome versions",
   ```

注：api调用时候内部数据交互依赖浏览器的`localstorage`。接口调用成功的用户信息都会存在localstorage.userData，所以代码必须在浏览器中运行，不能在nodejs环境运行。

API 接口变化说明

1. 1.4.6 开始引入呼叫号码加密功能，被叫号码加密方式和客户事先约定好。调用 call(number, callNumType) 传入加密号码和加密类型，sdk判断如果传入两个参数就认为被叫号码已加密，按约定方式解密；如果解密后结果包含非数字，认为解密失败，呼叫直接返回，返回结果是解密后字符串，便于调试查找问题。如果解密成功能正确呼出，呼叫回调的返回值里 data.n 依然是传入的加密字串。如果调用 call(number)只传一个参数按原来非加密方式处理。

2. 从 1.5开始 api接口 提供班长监控功能。**必须是班长权限的坐席调用这些接口才有实际效**果，普通坐席调用这些接口返回错误。使用班长监控基本分两步，具体使用详见下面api描述：
    1. 班长调用 init接口初始化并登录服务器的时候提供 `repCallStatus` 回调函数，用来接收本组坐席呼叫通知提醒。
    2. 调用各类监控相关接口需要提供两个参数，<1> 本监控呼叫的id <2> 监控结果的回调函数

    注：被监控的呼叫id我们称为 ccNumber， 以为call-id在sip信令协议中有别的含义。

API 接口描述

主要接口方法说明如下:

getUser(un, pwd, switchnumber,callback) 获取用户信息接口

注：该接口获取登录所需的所有参数 ， 是使用易米 webrtc api的第一步

请求参数说明

参数	是否必须	数据类型	参数说明
un	是	String	分机号
pwd	是	String	密码
switchnumber	是	String	企业总机号
callback	是	function(err,data) {}	请求成功 err 为 null ,data如下

返回数据 data 为 JSON 格式：

属性	属性说明
inGroups	[{id: "2033", eid: "65656", name: "售前咨询"},{...}]
userData	{displayname: "***", eid: "65656", number: "1023", ...}

err 为 JSON 格式，{code: xxx,info: 'xxxxxxxxxx'}

code	Info
50000	服务器错误
50001	参数错误
50002	获取运维地址失败
50003	分机号码或密码错误
50004	未找到该企业
50005	登录信息获取失败
其他	未知错误

调用示例

```
getUser('1**6', '1**6', '0256****734',function(err,data) {
    if(err) throw new Error('请求错误')
    // 可以从 data 中获取一些可用数据，比如：  eid,gid
})
```

getUser2({ un, pwd, switchnumber, callintype , number }, callback) 支持sip话机和回拨话机，获取用户信息接口

请求参数说明

un,pwd, switchnumber,callback 和 getUser 的含义相同，也都是必填参数

参数	是否必须	数据类型	参数说明
callintype	否	String	5 是sip话机，4回拨话机，2是原有的voip，其他输入返回错误
number	否	String	sip话机或回拨话机号码，如果callintype是4或5，number必填

从1.4.0 开始， api接口支持sip话机和回拨话机功能。使用它们，代码和之前**唯一不同**就是获取用户信息接口请调用`getUser2`. 原有接口`getUser`依然正常使用，但它只支持获取voip登录的相关信息。调用`getUser2`成功后，其他操作（登录、打电话）都和原来相同，所以代码不需要做别的改动。

用户拨打电话的界面操作也和原来相同，只是总机会回拨到sip话机（相比于之前回拨到浏览器）。用户可以在界面以及sip话机上挂断电话、通话保持。另有文档描述如何配置sip话机。

`getUser2` 采用es6对象解构赋值的方法传递参数，对象属性名称都是**小写**；调用传递参数如果用es6对象构建的简洁写法 Shorthand property names 注意变量名都要**小写**：

```
let un='1**6',pwd='1**6',switchnumber = '0256****734', callintype = 5, number = 'xxxx'
getUser2({un,pwd,switchnumber,callintype,number},function(err,data) {...})
```

详见 https://github.com/emicnet/test-jssip-emic 测试代码

init(params,eventCallback) 注册易米电话服务

注：

1. 调用 getUser 成功获取坐席信息(localstorage.userData 存入数据)就可以通过init初始化工具条。
2. init 注册登录易米服务器。当参数输入正确，init发起注册请求，返回true；当缺少必要参数，没法发起注册请求就返回false。
3. socketUri **在实际环境中必填**，不填会连到缺省的开发服务器。
4. 回调监听消息 （提供相应的回调接口才能正确使用易米电话服务）
    - 登录回调
    - 呼叫回调(呼出、呼入)
    - 被踢下线回调

参数	是否必须	数据类型	参数说明
params	是	Object	{ switchnumber，un ， pwd ...}
eventCallback	是	Object	{ register : function(){...},callEvent: function() {...} }

params

参数	是否必须	数据类型	参数说明
un	是	String	分机号
pwd	是	String	密码
switchnumber	是	String	总机号
gid	是	number	当前登陆技能组 id，如果不在任何技能组填0
remoteAudio	可选	string	当前播放语音的 auido 标签 id，没传使用缺省创建的 标签 id
socketUri	可选	url	网关地址。实用环境中必填（不然会连到开发服务器），比如 wss://webrtc01.emicloud.com:9060

eventCallback

参数	是否必须	数据类型	参数说明
register	是	function	注册回调 { code: 200, data: data } code 非 200 失败
callEvent	是	function	呼入会话监听
kickedOffLine	是	function	被踢 等下线消息（比如账号过期或者其他错误）监听 data
statusChanged	否	function	如果需要监听坐席状态可以设置这个回调
repCallStatus	否	function	班长登录提供该回调用于接收本组坐席呼叫通知, 1.5版本引入

回调通知说明

callEvent（type,data）

type	说明
callinResponse	座席外呼响应(内线)
calloutResponse	座席外呼响应(外线)
callinFaildResponse	座席外呼响应(内线) 被叫超时未接听
newPBXCall	来电呼入
cancelPBXCall	坐席拒接(会流转到其他空闲坐席) /客户挂断
answeredPBXCall	通话建立
endPBXCall	通话结束

data 格式说明

data	说明
a	sip 消息状态码
c	ccNumber
callTime	通话时长（只在 endPBXCall 时提供）
n	对方号码， calloutResponse/callinResponse/answeredPBXCall/newPBXCall 事件中带。 n的值是调用call(...) 传入值，所以从1.4.6开始可能加密后的值。

注：呼叫回调方法是使用api最复杂部分，可参见相应呼叫流程图和示例代码。

register(data)

data 格式说明

data	说明
code	200 登录成功，非200登录失败
info	登录失败原因说明
data	sip相关消息，便于sip信令调试，登录成功通常不需要处理

注：主动登出请调用 `logout()` 。登出正常情况下都是成功的；不成功通常是因为网络原因造成登出消息没送到服务器，但这种情况下坐席的实际状态通常也是下线的。 所以调用logout可以不提供回调，如果提供，回调参数和上述登录回调参数一样。

kickedOffLine(data)

data.r 是被踢下线原因：

data.r	说明
895	其他用户登录您的账号
897	注册超时，请重新登录
898	账号过期/账号被删/账号修改
898	企业停止使用

statusChanged(data)

data.status 0 离线，1在线，2 忙碌

和调用logout情况一样，设置坐席状态正常情况下都是成功的，所以这个回调函数非必须。设置状态不成功通常是因为网络原因造成登出消息没送到服务器，但这种情况下通常也没法收到回调。

但在首次注册时候可以通过这个回调确认坐席状态的变化（从离线变成在线）。因为服务器会在注册成功后马上给客户端发坐席在线的消息通知，正式通知坐席上线。

repCallStatus(data)

`repCallStatus` 是1.5版本引入班长监控功能增加的回调接口，非班长坐席即便提供了该回调也不会收到通知。同一个呼叫班长会收到**两次通知**，一次在呼叫发起时候，一次在呼叫结束时候。在班长登录前发起的呼叫不会有通知。

data 回调参数如下

data	说明
caller	主叫，即坐席分机号
ccNumber	呼叫唯一标识
callTime	呼叫发起时间，秒为单位的时间戳
endTime	呼叫结束时间，呼叫发起时候为 ''

班长获得呼叫的ccNumber后接着调用监控相关接口实施各种监控相关动作，详见监控相关接口

answerPBXCall(ccNumber) 来电接听

注：

1. callEvent 回调中 newPBXCall 可以拿到 ccnumber
2. 可在 callEvent 事件中监听通话不同状态

```
answerPBXCall('1564717108763414conf_1564717149723')
```

transferPBXCall(gid, transNumber, ccNumber, callback) 转接

请求参数说明

参数	是否必须	数据类型	参数说明
gid	是	string	技能组 id
tranNumber	是	string	分机号
ccNumber	是	string	当前通话唯一标识
callback	否	funtion(type,data) {...}	转接成功失败回调通知

调用实例

```
transferPBXCall(2069,1023,'1564717108763414conf_1564717149723',function(res){
    if(res.type=='transferCallSuccess'){
        //成功 res.data
    }
    if(res.type=='transferCallFaild'){
        //失败 res.data
    }
})
```

call(number, callNumType) 外呼

注：调用call发起呼叫后，相应的呼叫后续处理都在callEvent回调事件中。

请求参数说明

参数	是否必须	数据类型
number	是	String
callNumType	否	如果传值就认为 number 加密，按事先约定方式解密

返回值 2或3， 2表示呼叫外线（不用+9），3表示呼叫内线。调用实例

```
let calltype = phonebar.call('10010')
if (calltype ==2) console.log('呼叫外线')
// 1.4.6 加密呼叫示例
let res = phonebar.call('opwmRvGhOcGMXRZSp_moOQ', 'encrypt')
//如果解密不成功，解密后字串含非数字，res是解密后字串，call失败直接返回
//请参见演示示例，分别提供两种调用方式演示
```

呼叫不成功主要有以下原因（在`calloutResponse/callinResponse`）回调里处理，参见测试用例。

```
{

    '503': '对方忙碌',
    '507': '总机号已停机',
    '508': '非工作时间',
    '512': '该客户今日被呼叫次数已达上限',
    '1000': '禁止拨打无权限坐席'
}
```

terminate(ccNumber) 挂断

```
terminate('1564717108763414conf_1564717149723')
```

hold(ccNumber) 呼叫保持

unhold(ccNumber) 呼叫保持取消

logout(callback) 注销/退出

changeStaus(status) 修改坐席状态

```
changeStaus(1) // 1 空闲  2忙碌
```

checkLogin 获取登陆状态

```
var status = checkLogin() // 0 离线 1 在线
```

注，登录成功后马上调用这个方法返回0，因为服务器会在登录后才发出在线消息，参见示例。

checkSeatState 获取坐席状态

```
var status = checkSeatState() // 0 离线  1 空闲 2 忙碌(手动置忙，非通话中) 3 振铃 4 通话 5 保持
```

webApiHandler(functionName,webParam) 获取技能组/坐席（转接的时候需要）

注:

1. 当 getUser 调用成功后自动写入 `localstorage.userData` 数据, webApiHandler有些请求需要字段会从`localstorage`中获取。所以调用这两个方法之前要确保getUser 调用成功。
2. async/Promise 函数

请求参数说明

参数	是否必须	数据类型	参数说明
functionName	是	String	getGroups(获取所有技能组) ， searchEpMembers(获取技能组包含坐席)
webParam	是	Object	技能组 {un: '分机号',pwd: pwd:"密码",eid:企业 eid} 坐席(分页获取){un,pwd,eid,searchGid,length: 每页获取个数}

调用示例

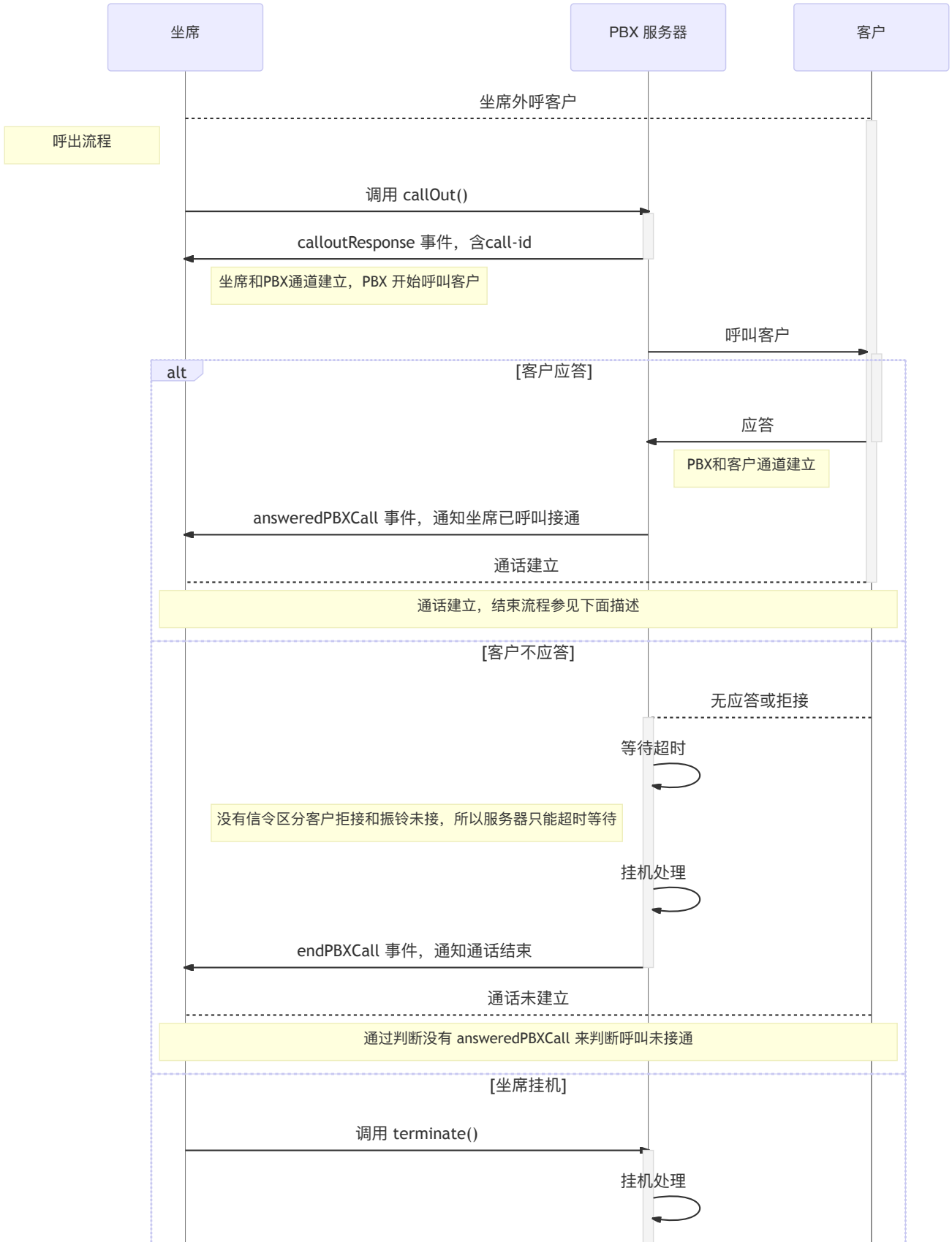
```
webApiHandler("getGroups",{
    un: 分机号,
```

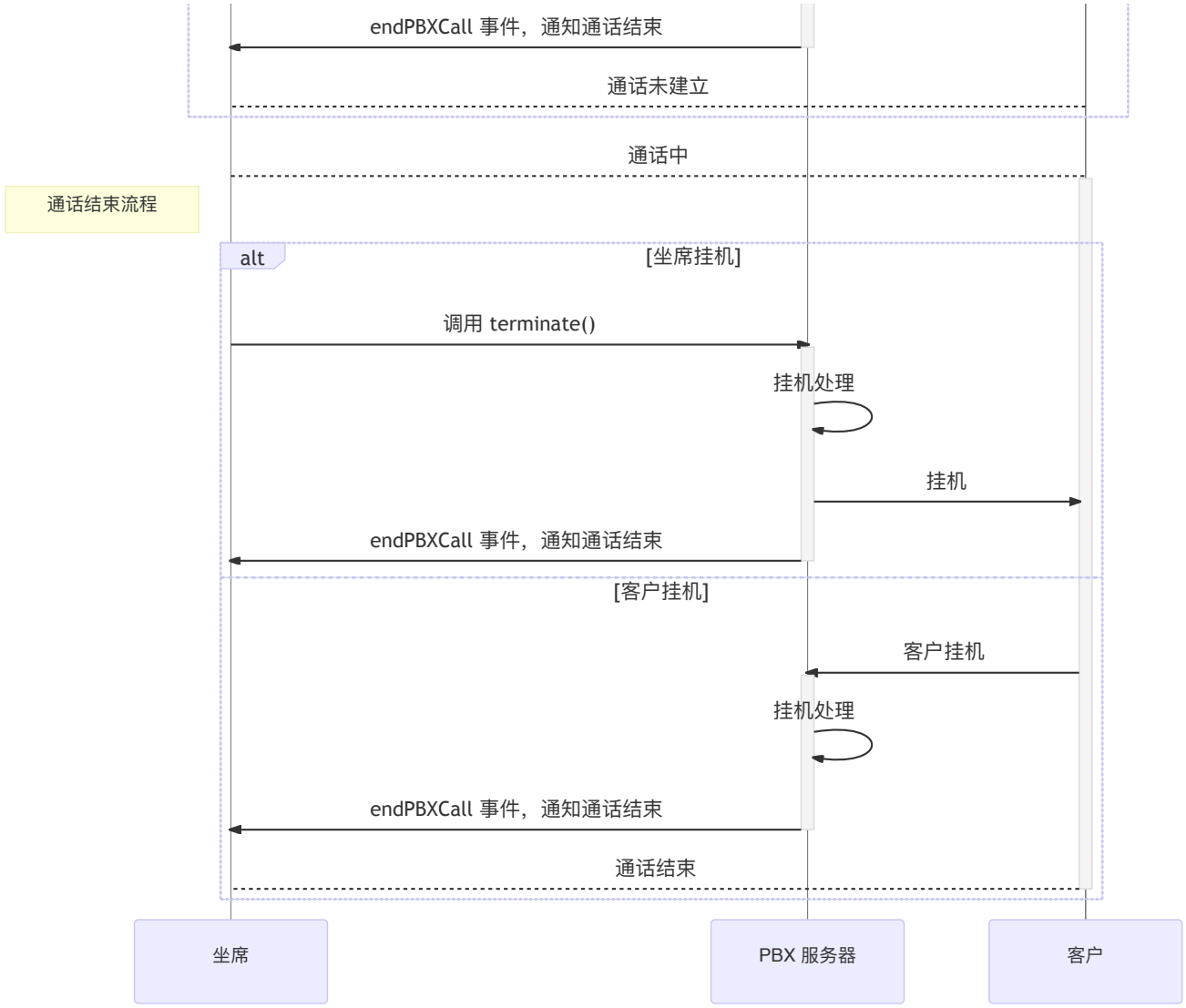
```
    pwd: 密码,
    eid: eid,
}).then(res=>{
    if(res.status==200){ }
    else { } //res.status = 50001 表示 localstorage.userData没有数据
})

let result = await webApiHandler ("searchEpMembers",{
    un: 分机号,
    pwd: 密码,
    eid: eid,
    searchGid: xxx技能组id,
    length: 10// 非必填
})
if (res.status == 200) {
    let number = res.returnData.recordsTotal //该组坐席总数
     for (const member of res.returnData.data) { //res.returnData.data是坐席数组
         //kefuStatus   0 离线  1 空闲  2 忙碌
         phonebar.log(`${member.displayname} 状态 ${member.kefuStatus}`)
     }
}
```

呼叫流程图

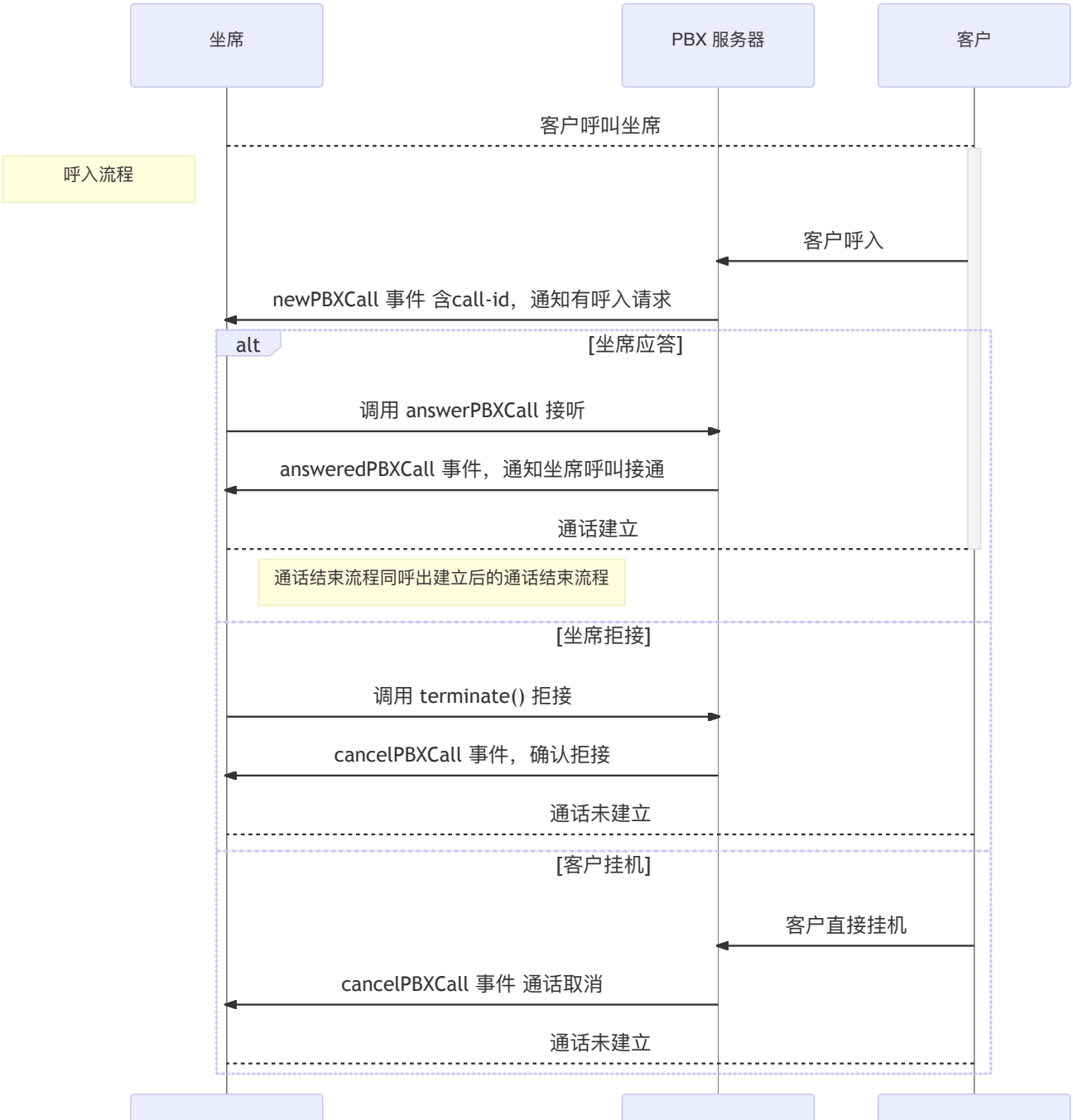
呼出

注：alt 表示几个不同处理分支


坐席
PBX 服务器
客户
坐席外呼客户
呼出流程
调用 callOut()
calloutResponse 事件，含call-id
坐席和PBX通道建立，PBX 开始呼叫客户
呼叫客户
alt
[客户应答]
应答
PBX和客户通道建立
answeredPBXCall 事件，通知坐席已呼叫接通
通话建立
通话建立，结束流程参见下面描述
[客户不应答]
无应答或拒接
等待超时
没有信令区分客户拒接和振铃未接，所以服务器只能超时等待
挂机处理
endPBXCall 事件，通知通话结束
通话未建立
通过判断没有 answeredPBXCall 来判断呼叫未接通
[坐席挂机]
调用 terminate()
挂机处理


endPBXCall 事件，通知通话结束
通话未建立
通话中
通话结束流程
alt
[坐席挂机]
调用 terminate()
挂机处理
挂机
endPBXCall 事件，通知通话结束
[客户挂机]
客户挂机
挂机处理
endPBXCall 事件，通知通话结束
通话结束
坐席
PBX 服务器
客户

呼入


坐席
PBX 服务器
客户
客户呼叫坐席
呼入流程
客户呼入
newPBXCall 事件 含call-id，通知有呼入请求
alt
[坐席应答]
调用 answerPBXCall 接听
answeredPBXCall 事件，通知坐席呼叫接通
通话建立
通话结束流程同呼出建立后的通话结束流程
[坐席拒接]
调用 terminate() 拒接
cancelPBXCall 事件，确认拒接
通话未建立
[客户挂机]
客户直接挂机
cancelPBXCall 事件 通话取消
通话未建立

基本示例实现外呼

```
phonebar.getUser("1002", "1002", '02566699734', function(err,data) {
    if (err) return
    var groups = data.inGroups //所属技能组
    var gid = groups && groups.length > 0 ? groups[0].id : 0
    //注册sip
    phonebar.init({
      switchnumber: "02566699734",
      gid: gid//登陆技能组
    }, {
      //注册回调
      register: function (data) {
        console.log("register", { data })
        if (data.code == 200) {
          //外呼
          phonebar.call('183*****136')
        }

      },
      //呼入通话回调
      callEvent: function (type, data) {
        console.log("callEvent", { type, data })
        if(type == 'newPBXCall') { // 来电呼入
           var ccNumber = data.c
           // 接听来电
           phonebar.answerPBXCall(ccNumber)
         }else if() {
              //...
         }
      },
      //被踢下线消息
      kickedOffLine: function (type, data) {
        console.log("kickedOffLine", { type, data })
```

```
      }

    })
})
```

监控相关接口 (1.5新增)

接口说明

要发起监控相关调用，需要先从`repCallStatus`回调接收坐席相关呼叫，取到相应 ccNumber， 再发起调用。监控操作是异步调用，在回调接口里返回结果，0为成功，非0（目前返回1）为失败:

```
let monitorcb = (result)=>{
   if (result ==0) {
     //监控调用成功
   } else {
     //监控调用失败
   }
}
```

startCallMonitor(ccNumber, monitorcb) 开始监听

要确保浏览器允许访问耳麦监听才能成功。 监听时候班长处在**静音模式**，他说话别人都听不到，坐席和客户感知不到有人在监听。

stopCallMonitor(ccNumber, monitorcb) 结束监听

结束监听，被监听通话不受影响，继续进行。

joinFromMonitor(ccNumber, monitorcb)

从监听状态进入三方通话。班长必须首先存在监听状态才能进入三方通话，如果不在监听状态直接调用该接口会失败。

exitJoinMonitor(ccNumber, monitorcb)

从三方通话退出，原有通话继续。

endCallMonitor(ccNumber, monitorcb)

从三方通话或监听状态中结束整个通话。不在监听或者三方通话中想要结束通话请调用`terminateCall`，此时调用`endCallMonitor`会失败。

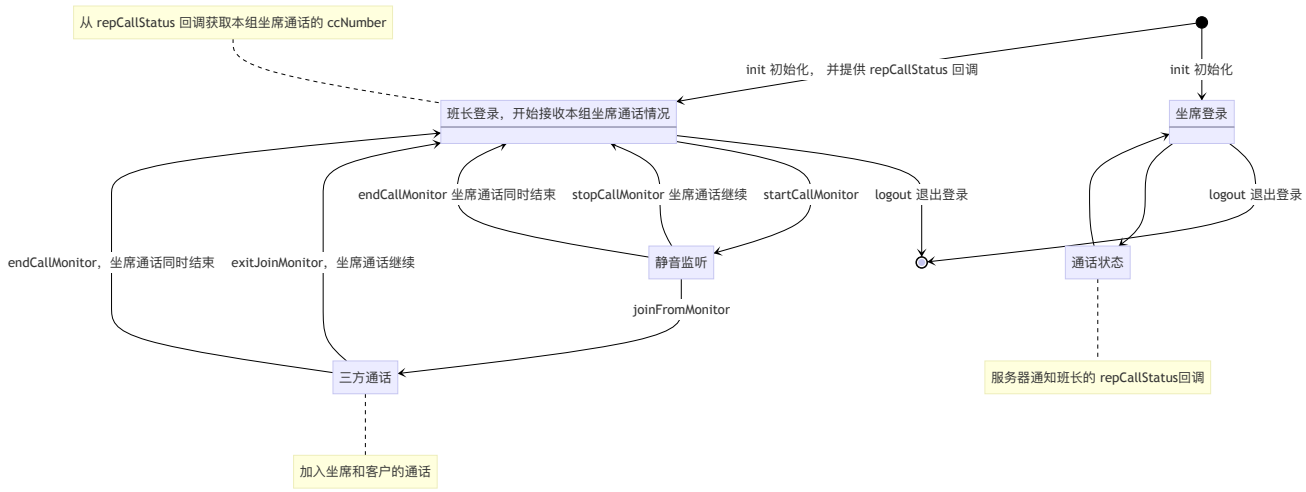
terminateCall(ccNumber, monitorcb)

不经过监听或者三方通话直接结束通话

checkMonitorState() 返回班长监控状态

```
{ status: 0, ccNumber: '' } //未监听
// 1 监听 2 监听转强插（即三方）
{ status: 1, ccNumber: 'xxx' }  //ccNumber由数字字母组成
```

班长监控状态图



监控示例代码

详细的监控代码请参见 https://github.com/emicnet/test-jssip-emic 示例，有几点提请注意：

1. 必须是班长才能对本技能组的坐席实施监听；普通坐席即便提供 repCallStatus的回调也不会收到通知。
2. 监听功能需要允许浏览器访问耳机和麦克风。chrome要求操作者手动授权这个安全权限，如果没有这

个权限监听会失败。

3. 监听时候会收到和呼叫相关的两个消息通知 (1) `answeredPBXCall` 表示通话建立，通过判断ccNumber可以知道这是监听通话；(2) 监听结束会收到 `endPBXCall` 表示通话结束，可以直接忽略。