AnyChat Server SDK

服务器应用程序编程接口

开发手册

（版本：V6.0）

广 州 佰 锐 网 络 科 技 有 限 公 司

Guangzhou BaiRui Network Technology Co.,Ltd.

http://www.bairuitech.com http://www.anychat.cn

2016 年 04 月

目 录

1. 概述 4

2. 回调函数定义 5

2.1. 服务器应用程序消息（扩展）回调函数定义 5
2.2. SDK定时器回调函数定义 6
2.3. 用户身份验证回调函数定义 6
2.4. 用户申请进入房间回调函数定义 7
2.5. 用户登录成功回调函数定义 8
2.6. 用户注销（扩展）回调函数定义 9
2.7. 用户进入房间回调函数定义 9
2.8. 用户离开房间回调函数定义 10
2.9. 上层业务自定义数据回调函数定义 10
2.10. 收到文字聊天数据回调函数定义 11
2.11. 透明通道数据回调函数 11
2.12. 透明通道数据扩展回调函数 12
2.13. 文件传输回调函数 12
2.14. 服务器录像回调函数 13
2.15. 视频呼叫事件回调函数 14
2.16. 用户信息控制回调函数 15

3. SDK接口规范 16

3.1. 设置服务器应用程序消息（扩展）回调函数 16
3.2. 设置SDK定时器回调函数 16
3.3. 设置用户身份验证回调函数 17
3.4. 设置用户申请进入房间回调函数 17
3.5. 设置用户登录成功回调函数 18
3.6. 设置用户注销（扩展）回调函数 18
3.7. 设置用户进入房间回调函数 18
3.8. 设置用户离开房间回调函数 19
3.9. 设置通信数据回调函数 19
3.10. 设置文字聊天数据回调函数 20
3.11. 设置透明通道数据回调函数 20
3.12. 设置透明通道数据扩展回调函数 21
3.13. 设置文件传输回调函数 21
3.14. 设置服务器录像通知回调函数 22
3.15. 设置视频呼叫事件回调函数 22
3.16. 设置用户信息事件回调函数 22
3.17. 设置回调函数 23
3.18. 获取SDK版本信息 23
3.19. 初始化SDK 24
3.20. 释放资源 24
3.21. 向指定用户发送数据 25

3.22. 向指定房间的所有用户发送数据.....25
3.23. 透明通道传送缓冲区.....26
3.24. 透明通道传送缓冲区扩展.....26
3.25. 传送文件.....27
3.26. 中心端录像控制.....28
3.27. 视频呼叫控制.....29
3.28. 设置用户的详细信息.....30
3.29. 获取用户的详细信息.....30
3.30. 用户信息控制.....31
4. 工作流程图.....32
5. 扩展数据传输通道.....33
5.1. 客户端之间使用透明通道传输数据.....33
5.2. 客户端之间使用透明通道扩展传输数据.....33
5.3. 客户端之间传输文件.....34
5.4. 客户端与服务器之间传输缓冲区数据（SDK FILTER DATA接口）.....34
5.5. 客户端与服务器之间传输缓冲区数据（透明通道接口）.....35
5.6. 客户端与服务器之间使用扩展缓冲区通道传输数据.....35
5.7. 客户端与服务器之间传输文件.....35
6. 视频呼叫业务.....37
6.1. 业务逻辑流程图.....37
6.2. API接口及常量定义.....37
6.3. 相关说明.....38
7. SDK使用与参数配置.....40
8. 常用业务处理逻辑.....41
8.1. IM中好友列表的实现.....41
8.2. IM中点对点聊天的实现.....41
8.3. 聊天室中公麦的实现.....42
8.4. 聊天室中麦序的实现.....43
9. 技术支持.....44

1. 概述

“AnyChat Server SDK”是 AnyChat 平台服务器端应用程序编程接口，用于实现 AnyChat 平台的可扩展应用，如业务逻辑的处理，也可用于 AnyChat 平台与第三方平台的互联互通。

我们通常将使用“AnyChat Server SDK”开发的应用程序称为业务层服务器，可以与 AnyChat 核心服务器程序部署于同一台计算机上，也可以部署在不同的计算机上，支持分布式部署。默认情况下（没有配置业务层服务器时），AnyChat 平台没有附带任何业务逻辑，如采用 AnyChat 开发会议室，或是聊天室时，里面的“麦序”、“公麦”等应用均属于业务层逻辑，需要由业务层服务器来处理，同时用自定义的指令，实现与客户端的数据交互，完成对应的业务逻辑。

“AnyChat Server SDK”与“SDK Filter Plus”均是服务器扩展编程接口，均为动态连接库（DLL）形式，两者的主要区别是：（1）、“SDK Filter Plus”的 DLL 被 AnyChat 核心服务器程序（AnyChatCoreServer.exe）调用，与 AnyChat 核心服务器程序属同一个进程；（2）、“AnyChat Server SDK”被业务层服务器程序（需要用户编写）调用，与 AnyChat 核心服务器程序属不同的进程，与 AnyChat 核心服务器采用 IPC 的方式进行通信。

“AnyChat Server SDK”与“SDK Filter Plus”两者可以实现相同的功能，通常来说，“SDK Filter Plus”适合业务逻辑较简单的应用，而“AnyChat Server SDK”则适合业务逻辑较复杂的应用，实现独立的业务层服务器，有对应的界面显示。

后面所有的描述，如无特殊说明，均是针对“AnyChat Server SDK”编程接口进行说明，有关“AnyChat 语音视频互动平台”的相关信息请参考《AnyChat 在线音视频互动平台开发手册》。

有关AnyChat平台用户身份验证与第三方平台集成的问题可参考技术论坛相关介绍：http://bbs.anychat.cn/forum.php?mod=viewthread&tid=12&extra=page%3D1

我们在 SDK 包中提供了“AnyChat Server SDK”的示例代码供参考，位于 SDK 包的 bin\serversdk 文件夹下面。

2. 回调函数定义

通过回调函数实现 SDK 底层的数据实时的反馈给上层应用。回调函数的最后一个参数均为：LPVOID lpUserValue，可供用户自定义，默认为 NULL，通常上层应用可传递一个对象的句柄（指针），在回调函数体里面再通过强制类型转换，变为上层应用可用的对象句柄（指针），详细用法可参考示例程序中回调函数相关部分。

2.1. 服务器应用程序消息（扩展）回调函数定义

typedef void (CALLBACK* BRAS_OnServerAppMessageEx_CallBack)(DWORD dwMsg, DWORD wParam, DWORD lParam, LPVOID lpUserValue);

参数：

DWORD	dwMsg：	应用程序消息类型；
DWORD	wParam：	附带参数
DWORD	lParam；	附带参数
LPVOID	lpUserValue：	用户自定义参数，在设置回调函数时传入

备注：

当“AnyChat Server SDK”与AnyChat平台核心服务器建立IPC通信，或是IPC通信中断（如核心服务器被关闭）时，将触发该回调函数，上层应用可根据dwMsg（消息类型）参数来进行对应的处理，如收到与AnyChat服务器断开的消息时，业务层服务器可以清除对应的内存数据，恢复所有状态为初始状态，这样便可保证业务层服务器与核心服务器之间的数据，或是状态是一致的。

目前已定义的应用程序消息类型及其附带参数定义如下：

```
// 与核心服务器的连接消息，wParam为errorcode
#define BRAS_MESSAGE_CORESERVERCONN        10
// 与录像服务器的连接消息，wParam为errorcode，lParam为recordserverid
#define BRAS_MESSAGE_RECORDSERVERCONN      11
// 与登录服务器的连接消息，wParam为errorcode，lParam为loginserverid
#define BRAS_MESSAGE_LOGINSERVERCONN       12
// 与房间服务器的连接消息，wParam为errorcode，lParam为roomserverid
```

```
#define BRAS_MESSAGE_ROOMSERVERCONN        13
// 与流媒体服务器的连接消息，wParam为errorcode，lParam为mediaserverid
#define BRAS_MESSAGE_MEDIASERVERCONN       14
```

2.2. SDK定时器回调函数定义

typedef void (CALLBACK* BRAS_OnTimerEvent_CallBack)(LPVOID lpUserValue);

参数：

LPVOID lpUserValue： 用户自定义参数，在设置回调函数时传入

备注：

该回调函数将定期被触发，触发间隔由设置回调函数时的参数决定。

上层应用可以在该回调中处理定时任务，而不需要额外开启线程，或是定时器，由于该回调函数与其它回调函数均是顺序被触发，所以在该回调函数里面可以不用考虑同步的问题，而如果上层应用单独开启线程来完成相关的任务，则必须考虑数据之间多线程同步的问题。

2.3. 用户身份验证回调函数定义

typedef DWORD (CALLBACK* BRAS_VerifyUser_CallBack)(IN LPCTSTR lpUserName,IN LPCTSTR lpPassword, OUT LPDWORD lpUserID, OUT LPDWORD lpUserLevel, OUT LPTSTR lpNickName,IN DWORD dwNCLen, LPVOID lpUserValue);

参数：

LPCTSTR	lpUserName	用于身份验证的用户名
LPCTSTR	lpPassword	用于身份验证的用户密码
LPDWORD	lpUserID	输出用户的 ID 号（唯一）
LPDWORD	lpUserLevel	输出用户的级别
LPTSTR	lpNickName	输出用户的昵称（可以与用户名相同）
DWORD	dwNCLen	保存用户昵称的缓冲区大小
LPVOID	lpUserValue	用户自定义参数，在设置回调函数时传入

返回值：

DWORD 返回出错代码，参考 SDK\ GVErrorCodeDefine.h 文件中有关出错代码的定义，通常会返回如下的值：

GV_ERR_SUCCESS（0） 表示验证通过，允许用户登录系统

GV_ERR_CERTIFY_FAIL(200) 用户名或密码有误，认证失败

GV_ERR_ACCOUNT_LOCK(202) 帐户已被暂时锁定，认证失败

备注：

该回调函数用于验证用户的身份信息。用户名与用户密码为输入参数，后面其它的参数为输出参数，用于反馈结果给服务器程序。身份验证是否成功通过该方法的返回值来判定。dwVSLen 用于指示传入的缓冲区大小，防止函数体里面拷贝数据时内存溢出。如果该方法不输出用户的昵称，即当 lpNickName 的字符长度为 0 时，服务器会自动将用户名默认为用户的昵称。

当客户端调用 API:“BRAC_Login”方法，且用户密码不为空时将会触发该回调函数，如果用户密码为空，则不会触发该回调，服务器会将该客户端当“游客身份”进行处理，如果服务器不支持游客方式登录，则将会返回对应的出错代码给客户端。

该回调仅仅做为用户身份验证之用，不能在该回调里面调用其它的 API（如向指定用户传输透明通道数据：BRAS_TransBuf 等），因为该回调没有返回之前，用户在内核就没有与 USERID 进行绑定，所以这时调用其它的 API 将是无效的，如需要在用户登录之后就发送一些数据，则可在 BRAS_OnUserLoginAction_CallBack 回调函数里面进行处理。

2.4. 用户申请进入房间回调函数定义

typedef DWORD (CALLBACK* BRAS_PrepareEnterRoom_CallBack)(DWORD dwUserId, DWORD dwRoomId, LPCTSTR lpRoomName,LPCTSTR lpPassword, LPVOID lpUserValue);

参数：

DWORD	dwUserId	请求者用户 ID
DWORD	dwRoomId	请求进入的房间编号
LPCTSTR	lpRoomName	请求进入的房间名

LPCTSTR	lpPassword	房间密码
LPVOID	lpUserValue	用户自定义参数，在设置回调函数时传入

返回值：

DWORD 返回出错代码，参考 SDK\ GVErrorCodeDefine.h 文件中有关出错代码的定义。

备注：

该回调函数将在客户端 SDK 调用进入房间的相关 API 接口后将被触发，业务层服务器需要判断该请求的合法性，服务器根据返回值决定是否让用户进入房间。

当客户端调用 API“BRAC_EnterRoom”，根据房间 ID 进入房间时，dwRoomId 与 lpPassword 参数有效。

当客户端调用 API“BRAC_EnterRoomEx”，根据房间名进入房间时，lpRoomName 与 lpPassword 参数有效，此时的 dwRoomId 是由服务器自动分配的。

2.5. 用户登录成功回调函数定义

typedef void (CALLBACK* BRAS_OnUserLoginAction_CallBack)(DWORD dwUserId, LPCTSTR szUserName, DWORD dwLevel, LPCTSTR szIpAddr, LPVOID lpUserValue);

参数：

DWORD	dwUserId	登录用户 ID
LPCTSTR	szUserName	登录用户名
DWORD	dwLevel	登录用户级别
LPCTSTR	szIpAddr	登录用户 IP 地址
LPVOID	lpUserValue	用户自定义参数，在设置回调函数时传入

备注：

当用户登录系统成功之后，SDK 内部会触发该回调函数，通知业务层服务器有用户登录，业务层服务器可将该用户的信息记录下来，然后根据不同的应用进行对应的处理，如 IM 应用中，当用户登录成功之后，可将用户的好友及其在线状态发送给该用户（需要业务层自定义指令及相关的数据结构），客户端收到业

务层服务器发送的好友列表数据后，便可显示好友列表及其在线状态了。

2.6. 用户注销（扩展）回调函数定义

typedef void (CALLBACK* BRAS_OnUserLogoutActionEx_CallBack)(DWORD dwUserId, DWORD dwErrorCode, LPVOID lpUserValue);

参数：

DWORD	dwUserId	注销用户的 ID
DWORD	dwErrorCode	出错代码，表示用户注销（离线）的原因
LPVOID	lpUserValue	用户自定义参数，在设置回调函数时传入

备注：

当用户从系统注销（调用 API“BRAC_Logout”），或是网络断开之时，SDK 内部会触发该回调函数，通知业务层服务器有用户离开系统，这时业务层服务器可进行相应的逻辑处理，如 IM 应用中，给该用户的在线好友发送用户离开指令，告知该用户已下线。

业务服务器可根据 dwErrorCode 所表示的出错代码来判断用户离线的原因，通常有：

dwErrorCode：0	表示客户端正常离线；
dwErrorCode：100	表示客户端网络通信超时所导致的被动离线；
dwErrorCode：101	表示客户端网络异常所导致的被动离线；
dwErrorCode：211	表示被服务器端踢掉所导致的被动离线；

2.7. 用户进入房间回调函数定义

typedef void (CALLBACK* BRAS_OnUserEnterRoomAction_CallBack)(DWORD dwUserId, DWORD dwRoomId, LPVOID lpUserValue);

参数：

DWORD	dwUserId	进入房间用户的 ID
DWORD	dwRoomId	进入的房间 ID

LPVOID	lpUserValue	用户自定义参数，在设置回调函数时传入

备注：

当用户调用 API“BRAC_EnterRoom”进入房间成功（在 PrepareEnterRoom 回调中返回 0）之后，SDK 会触发该回调函数，通知业务层服务器有用户进入房间。

2.8. 用户离开房间回调函数定义

typedef void (CALLBACK* BRAS_OnUserLeaveRoomAction_CallBack)(DWORD dwUserId, DWORD dwRoomId, LPVOID lpUserValue);

参数：

DWORD	dwUserId	离开房间用户的 ID
DWORD	dwRoomId	离开的房间 ID
LPVOID	lpUserValue	用户自定义参数，在设置回调函数时传入

备注：

当用户调用 API“BRAC_LeaveRoom”离开房间，或是网络异常断开之时，SDK 会触发该回调函数，通知业务层服务器有用户离开房间。

2.9. 上层业务自定义数据回调函数定义

typedef void (CALLBACK* BRAS_OnRecvUserFilterData_CallBack)(DWORD dwUserId, LPCSTR lpBuf, DWORD dwLen, LPVOID lpUserValue);

参数：

DWORD	dwUserId	发送数据的用户 ID
LPCSTR	lpBuf	数据缓冲区
DWORD	dwLen	缓冲区大小

备注：

当客户端调用“BRAC_SendSDKFilterData”方法向服务器发送数据时，服务器收到数据后，SDK 会触发该回调函数，把客户端发送的数据交给业务层服务器来处理，缓冲区内容不限，可以为任意类型数据，业务层可在该回调函数里面

分析缓冲区数据，进行自定义指令的解析，实现业务的处理。

2.10. 收到文字聊天数据回调函数定义

typedef void (CALLBACK* BRAS_OnRecvUserTextMsg_CallBack)(DWORD dwRoomId, DWORD dwSrcUserId, DWORD dwTarUserId, BOOL bSecret, LPCTSTR lpTextMessage, DWORD dwLen, LPVOID lpUserValue);

参数：

DWORD	dwRoomId	文字消息所对应的房间编号
DWORD	dwSrcUserId	源用户 ID，文字消息发送者
DWORD	dwTarUserId	目标用户 ID，消息接收者，-1 表示所有人
BOOL	bSecret	是否为悄悄话，目标用户 ID 不为-1 时有效
LPCTSTR	lpTextMessage	文字消息内容
DWORD	dwLen	文字消息长度

备注：

当客户端调用“BRAC_SendTextMessage”向房间内用户发送文字消息时，SDK 将会触发该回调函数。

在某些应用场合中，需要在服务器上保存用户的所有文字聊天记录，则可通过该回调函数来完成，如可将相关内容写入数据库中。

2.11. 透明通道数据回调函数

typedef void (CALLBACK * BRAS_TransBuffer_CallBack)(DWORD dwUserid, LPBYTE lpBuf, DWORD dwLen, LPVOID lpUserValue);

参数：

dwUserid：	用户 ID，指示发送用户
lpBuf：	缓冲区地址
dwLen：	缓冲区大小
lpUserValue：	用户自定义参数，在设置回调函数时传入

备注：

当收到客户端使用“BRAC_TransBuffer”方法发送的数据，且目标用户 ID 为 0 时，服务器将会触发该回调函数。

由于该函数传递的数据是一个与本 SDK 无关的缓冲区（由上层应用自己填充内容），相对于本 SDK 来说是透明的，故称为透明通道，利用该通道，可以向当前房间内的任何用户传输上层应用自定义的数据。

2.12. 透明通道数据扩展回调函数

typedef void (CALLBACK * BRAS_TransBufferEx_CallBack)(DWORD dwUserid, LPBYTE lpBuf, DWORD dwLen, DWORD wParam, DWORD lParam, DWORD dwTaskId, LPVOID lpUserValue);

参数：

dwUserid：	用户 ID，指示发送用户
lpBuf：	缓冲区地址
dwLen：	缓冲区大小
wParam：	缓冲区附带参数（由发送者设置，上层应用可自定义用途）
lParam：	缓冲区附带参数 2
dwTaskId：	该缓冲区所对应的传输任务编号
lpUserValue：	用户自定义参数，在设置回调函数时传入

备注：

当收到客户端使用“BRAC_TransBufferEx”方法发送的数据（目标用户为 0）时，将会触发该回调函数。

2.13. 文件传输回调函数

typedef void (CALLBACK * BRAS_TransFile_CallBack)(DWORD dwUserid, LPCTSTR lpFileName, LPCTSTR lpTempFilePath, DWORD dwFileLength, DWORD wParam, DWORD lParam, DWORD dwTaskId, LPVOID lpUserValue);

参数：

dwUserid： 用户 ID，指示发送用户

lpFileName： 文件名（含扩展名，不含路径）

lpTempFilePath：接收完成后，SDK 保存在本地的临时文件（包含完整路径）

dwFileLength： 文件总长度

wParam： 附带参数 1

lParam： 附带参数 2

dwTaskId： 该文件所对应的任务编号

lpUserValue： 用户自定义参数，在设置回调函数时传入

备注：

当收到客户端使用“BRAC_TransFile”方法发送的文件（目标用户为 0）时，将会触发该回调函数。 服务器收到文件后，将保存在配置文件所配置的临时文件路径下，然后再回调给上层应用。

特别提示：本 SDK 不会删除“lpTempFilePath”所指示的临时文件，上层应用在处理完毕后，需要主动删除该临时文件。

2.14. 服务器录像回调函数

typedef void (CALLBACK * BRAS_OnServerRecord_CallBack)(DWORD dwUserId, LPCTSTR lpFileName, DWORD dwElapse, DWORD dwFlags, DWORD dwParam, LPCTSTR lpUserStr, DWORD dwRecordServerId, LPVOID lpUserValue);

参数：

dwUserId	用户 ID，指示录像文件所对应的用户编号
lpFileName	文件名（含路径）
dwElapse	录像时长，单位：秒
dwFlags	附带标志
dwParam	附带参数（整形）
lpUserStr	附带参数（字符串类型）
dwRecordServerId	录像服务器 ID（保留）

lpUserValue： 回调函数自定义参数，在设置回调函数时传入

备注：

AnyChat Server SDK 并不负责音视频录制，而是由专门的中心录像服务器完成，中心录像服务器与核心服务器建立连接，上层应用调用 AnyChat Server SDK 所提供的 API 只是向录像服务器下达开始（或停止）的录像指令，当录像成功之后，将会触发该回调函数。

录像文件名（含路径），可以是相对路径，绝对路径，或是URL路径，在中心录像服务器参数配置文件中AnyChatRecordServer.ini中配置，可参考：中心录像服务器返回录像文件路径可配置

在服务器端录制音视频，需要单独部署中心录像服务器，参考： Windows平台中心录像服务器部署、Linux平台中心录像服务器部署

更多信息可参考：

AnyChat支持录像文件格式设置（MP4、WMV、FLV、MP3）

中心服务器录像支持触发客户端回调事件

2.15. 视频呼叫事件回调函数

typedef DWORD (CALLBACK * BRAS_OnVideoCallEvent_CallBack)(DWORD dwEventType, DWORD dwSrcUserId, DWORD dwTarUserId, DWORD dwErrorCode, DWORD dwFlags, DWORD dwParam, LPCTSTR lpUserStr, LPVOID lpUserValue);

参数：

dwEventType 事件类型，参考“视频呼叫业务”章节相关定义；

dwSrcUserId 呼叫发起方用户 ID；

dwTarUserId 被呼叫方用户 ID；

dwErrorCode 出错代码；

dwFlags 呼叫标志，参考“视频呼叫业务”章节相关定义；

dwParam 附带参数（整型），用户自定义；

lpUserStr 附带参数（字符串），用户自定义；

lpUserValue： 用户自定义参数，在设置回调函数时传入；

备注：

dwParam、lpUserStr 均为用户自定义参数，在客户端调用 API（BRAC_VideoCallControl）时传入。

当 dwEventType 为呼叫请求（BRAC_VIDEOCALL_EVENT_REQUEST）时，返回 0 表示允许当前请求，否则表示拒绝当前请求，返回值为出错代码。将体现在客户端的回复（Reply）事件中。

当 dwEventType 为呼叫开始（BRAC_VIDEOCALL_EVENT_START）时，dwParam 表示房间号（RoomId），由核心服务器统一分配。

有关视频呼叫业务可参考“视频呼叫业务”章节，还可以参考论坛技术文档：http://bbs.anychat.cn/forum.php?mod=viewthread&tid=150&extra=page%3D1。

2.16. 用户信息控制回调函数

typedef DWORD (CALLBACK * BRAS_OnUserInfoControl_CallBack)(DWORD dwSendUserId, DWORD dwUserId, DWORD dwCtrlCode, DWORD wParam, DWORD lParam, LPCTSTR lpStrValue, LPVOID lpUserValue);

参数：

dwSrcUserId	控制指令发起方用户 ID；
dwUserId	目标用户 ID（操作对象）；
dwCtrlCode	控制代码；
wParam	附带参数（整型），用户自定义；
lParam	附带参数（整型），用户自定义；
lpStrValue	附带参数（字符串），用户自定义；
lpUserValue：	用户自定义参数，在设置回调函数时传入；

备注：

wParam、lParam、lpStrValue 均为用户自定义参数，在客户端调用 API（BRAC_UserInfoControl）时传入。

3. SDK接口规范

“AnyChat Server SDK”采用“**C 调用约定**”（即用**__cdecl** 关键字说明），_cdecl 是 C 和 C++程序的缺省调用方式，如果是采用其它语言（如 VB.Net、C#、Delphi 等）调用 SDK 编写业务层服务器，则需要注意是否需要申明该调用方式。

3.1. 设置服务器应用程序消息（扩展）回调函数

DWORD

BRAS_SetOnServerAppMessageExCallBack(BRAS_OnServerAppMessageEx_CallBack lpFunction, LPVOID lpUserValue);

功能：设置服务器应用程序消息（扩展）回调函数，当该回调函数触发条件满足时，将会被 SDK 所触发。

返回值：0 表示设置成功，否则为出错代码

参数：

lpFunction	服务器应用程序消息（扩展）回调函数地址
lpUserValue	用户自定义参数，在回调函数中返回给上层应用

3.2. 设置SDK定时器回调函数

DWORD BRAS_SetTimerEventCallBack(DWORD dwElapse, BRAS_OnTimerEvent_CallBack lpFunction, LPVOID lpUserValue);

功能：设置 SDK 定时器回调函数，SDK 将定期触发该回调函数。

返回值：0 表示设置成功，否则为出错代码

参数：

dwElapse	定时器间隔，单位：ms
lpFunction	SDK 定时器回调函数地址
lpUserValue	用户自定义参数，在回调函数中返回给上层应用

3.3. 设置用户身份验证回调函数

DWORD BRAS_SetVerifyUserCallBack(BRAS_VerifyUser_CallBack lpFunction, LPVOID lpUserValue);

功能：设置用户身份验证回调函数，当客户端调用 API“BRAC_Login”，且密码参数不为空时，SDK 将触发该回调函数，让业务层服务器来实现用户身份的验证。

返回值：0 表示设置成功，否则为出错代码

参数：

lpFunction 用户身份验证回调函数地址

lpUserValue 用户自定义参数，在回调函数中返回给上层应用

备注：

SDK 将根据回调函数返回值来确定登录是否成功，如果验证成功，SDK 将触发“OnUserLoginActionCallBack”回调函数。

3.4. 设置用户申请进入房间回调函数

DWORD BRAS_SetPrepareEnterRoomCallBack(BRAS_PrepareEnterRoom_CallBack lpFunction, LPVOID lpUserValue);

功能：设置用户申请进入房间回调函数，当客户端调用 API“BRAC_EnterRoom”时，SDK 将触发该回调函数，让业务层服务器来确定是否允许该用户进入指定的房间。

返回值：0 表示设置成功，否则为出错代码

参数：

lpFunction 用户申请进入房间回调函数地址

lpUserValue 用户自定义参数，在回调函数中返回给上层应用

备注：

SDK 将根据回调函数返回值来确定进入房间是否成功，如果成功，SDK 将触发“OnUserEnterRoomActionCallBack”回调函数。

3.5. 设置用户登录成功回调函数

DWORD BRAS_SetOnUserLoginActionCallBack(BRAS_OnUserLoginAction_CallBack lpFunction, LPVOID lpUserValue);

功能：设置用户登录成功回调函数，当有用户登录系统成功时，所设置的回调函数将被触发。

返回值：0 表示设置成功，否则为出错代码

参数：

lpFunction	用户登录成功回调函数地址
lpUserValue	用户自定义参数，在回调函数中返回给上层应用

3.6. 设置用户注销（扩展）回调函数

DWORD BRAS_SetOnUserLogoutActionExCallBack(BRAS_OnUserLogoutActionEx_CallBack lpFunction, LPVOID lpUserValue);

功能：设置用户注销（扩展）回调函数，当有用户登录系统成功之后，调用 API “BRAC_Logout”时，所设置的回调函数将被触发。

返回值：0 表示设置成功，否则为出错代码

参数：

lpFunction	用户注销回调函数地址
lpUserValue	用户自定义参数，在回调函数中返回给上层应用

备注：

当已登录成功的用户网络异常断开时，SDK 已将会触发所设置的回调函数。

3.7. 设置用户进入房间回调函数

DWORD BRAS_SetOnUserEnterRoomActionCallBack(BRAS_OnUserEnterRoomAction_CallBack lpFunction, LPVOID lpUserValue);

功能：设置用户进入房间回调函数，当有用户进入房间成功时，所设置的回调函数将被触发。

返回值：0 表示设置成功，否则为出错代码

参数：

lpFunction 用户进入房间回调函数地址

lpUserValue 用户自定义参数，在回调函数中返回给上层应用

3.8. 设置用户离开房间回调函数

DWORD BRAS_SetOnUserLeaveRoomActionCallBack(BRAS_OnUserLeaveRoomAction_CallBack lpFunction, LPVOID lpUserValue);

功能：设置用户离开房间回调函数，当有用户进入房间成功之后，调用 API “BRAC_LeaveRoom”时，所设置的回调函数将被触发。

返回值：0 表示设置成功，否则为出错代码

参数：

lpFunction 用户离开房间回调函数地址

lpUserValue 用户自定义参数，在回调函数中返回给上层应用

备注：

当已登录成功且进入房间的用户网络异常断开时，SDK 已将会触发所设置的回调函数。

3.9. 设置通信数据回调函数

DWORD BRAS_SetOnRecvUserFilterDataCallBack(BRAS_OnRecvUserFilterData_CallBack lpFunction, LPVOID lpUserValue);

功能：设置通信数据回调函数，当用户登录系统成功之后，调用 API “BRAC_SendSDKFilterData”向服务器发送数据时，所设置的回调函数将被触发。

返回值：0 表示设置成功，否则为出错代码

参数：

lpFunction 通信数据回调函数地址

lpUserValue 用户自定义参数，在回调函数中返回给上层应用

备注：

当客户端向服务器发送自定义通信数据时，服务器将触发所设置的回调函数，数据内容可由上层应用自己定义，可以为任何数据类型，业务层服务器可在回调函数中分析所收到的缓冲区内容并进行对应的处理。

3.10. 设置文字聊天数据回调函数

DWORD BRAS_SetOnRecvUserTextMsgCallBack(BRAS_OnRecvUserTextMsg_CallBack lpFunction, LPVOID lpUserValue);

功能：设置文字交互数据回调函数，当用户进入房间成功，并调用 API “BRAC_SendTextMessage”向房间内用户发送文字聊天信息时，所设置的回调函数将被触发。

返回值：0 表示设置成功，否则为出错代码

参数：

lpFunction 文字交互数据回调函数地址

lpUserValue 用户自定义参数，在回调函数中返回给上层应用

备注：

通常应用于需要保存文字聊天记录的需求。

3.11. 设置透明通道数据回调函数

DWORD BRAS_SetTransBufferCallBack(BRAC_TransBuffer_CallBack lpFunction, LPVOID lpUserValue);

功能：设置透明通道数据回调函数，使得当收到客户端发送的透明通道数据时，能通过回调函数，将透明通道的缓冲区数据回调给上层应用。

返回值：0 表示成功，否则为出错代码

参数：

lpFunction 回调函数地址，函数定义参考“回调函数”一节；

lpUserValue 用户自定义参数，该参数在回调函数中被返回，默认为 NULL，通常传入一个对象的地址（指针）。

3.12. 设置透明通道数据扩展回调函数

DWORD BRAS_SetTransBufferExCallBack (BRAC_TransBufferEx_CallBack lpFunction, LPVOID lpUserValue);

功能：设置透明通道数据扩展回调函数，使得当收到客户端通过透明通道扩展函数发送数据时，能通过指定的回调函数，将透明通道的缓冲区数据回调给上层应用。

返回值：0 表示成功，否则为出错代码

参数：

lpFunction 回调函数地址，函数定义参考“回调函数”一节；

lpUserValue 用户自定义参数，该参数在回调函数中被返回，默认为 NULL，通常传入一个对象的地址（指针）。

3.13. 设置文件传输回调函数

BRAC_API DWORD BRAS_SetTransFileCallBack(BRAC_TransFile_CallBack lpFunction, LPVOID lpUserValue);

功能：设置文件传输回调函数，使得当有客户端通过文件传输函数发送文件，本地接收成功后，能通过指定的回调函数，将文件名、本地临时文件路径、文件大小等数据回调给上层应用。

返回值：0 表示成功，否则为出错代码

参数：

lpFunction 回调函数地址，函数定义参考“回调函数”一节；

lpUserValue 用户自定义参数，该参数在回调函数中被返回，默认为 NULL，通常传入一个对象的地址（指针）。

3.14. 设置服务器录像通知回调函数

DWORD BRAS_SetOnServerRecordCallBack(BRAS_OnServerRecord_CallBack lpFunction, LPVOID lpUserValue);

功能：设置服务器录像通知回调函数，当中心录像服务器完成录像任务后，将触发上层应用设置的回调函数，上层应用可在回调函数中对录像文件进行集中管理。

返回值：0 表示成功，否则为出错代码

参数：

lpFunction 回调函数地址，函数定义参考“回调函数”一节；

lpUserValue 用户自定义参数，该参数在回调函数中被返回，默认为 NULL，通常传入一个对象的地址（指针）。

3.15. 设置视频呼叫事件回调函数

DWORD BRAS_SetOnVideoCallEventCallBack(BRAS_OnVideoCallEvent_CallBack lpFunction, LPVOID lpUserValue);

功能：设置视频呼叫事件回调函数，当客户端发起视频呼叫业务流程，调用 API：BRAC_VideoCallControl 时将触发上层应用设置的回调函数，上层应用可在回调函数中对呼叫业务流程进行管理。

返回值：0 表示成功，否则为出错代码

参数：

lpFunction 回调函数地址，函数定义参考“回调函数”一节；

lpUserValue 用户自定义参数，该参数在回调函数中被返回，默认为 NULL，通常传入一个对象的地址（指针）。

3.16. 设置用户信息事件回调函数

DWORD BRAS_SetOnUserInfoControlCallBack(BRAS_OnUserInfoControl_CallBack lpFunction, LPVOID lpUserValue);

功能：设置用户信息控制事件回调函数，当客户端发起用户信息控制，调用 API：BRAC_UserInfoControl 时将触发上层应用设置的回调函数，上层应用可在回调函数中对用户信息进行处理。

返回值：0 表示成功，否则为出错代码

参数：

lpFunction 回调函数地址，函数定义参考“回调函数”一节；

lpUserValue 用户自定义参数，该参数在回调函数中被返回，默认为 NULL，通常传入一个对象的地址（指针）。

3.17. 设置回调函数

DWORD BRAS_SetCallBack(DWORD dwCBType, LPVOID lpFunction, LPVOID lpUserValue);

功能：设置回调函数。

返回值：0 表示成功，否则为出错代码

参数：

dwCBType 回调函数类型，定义为常量：BRAS_CBTYPE_XXXX

lpFunction 回调函数地址，函数定义参考“回调函数”一节；

lpUserValue 用户自定义参数，该参数在回调函数中被返回，默认为 NULL，通常传入一个对象的地址（指针）。

备注：

该方法可以设置所有回调函数。

3.18. 获取SDK版本信息

DWORD BRAS_GetSDKVersion(DWORD& dwMainVer, DWORD& dwSubVer, LPSTR lpCompileTime, DWORD dwBufLen);

功能：获取 SDK 版本信息以及编译时间等信息。

返回值：0 表示成功，否则为出错代码

参数：

dwMainVer： 返回 SDK 的主版本号

dwSubVer： 返回 SDK 的从版本号

lpCompileTime：返回 SDK 的编译时间

dwBufLen： 保存编译时间信息缓冲区的大小

备注：

通过调用该方法可以获得 SDK 的版本及编译时间等信息，在遇到版本不兼容问题时，有助于判断故障原因。

3.19. 初始化SDK

DWORD BRAS_InitSDK(DWORD dwReserved);

功能：初始化服务器 SDK。

返回值：0 表示成功，否则为出错代码

参数：

dwReserved 保留

备注：

只有 SDK 初始化完成后，才能正常工作。

3.20. 释放资源

DWORD BRAS_Release(void);

功能：释放服务器 SDK 所占用的资源。

返回值：0 表示成功，否则为出错代码

参数：

void

备注：

在业务层服务器退出时，必须释放资源，否则可能导致异常。

3.21. 向指定用户发送数据

DWORD BRAS_SendBufToUser(DWORD dwUserId, LPCTSTR lpBuf, DWORD dwLen);

功能：向指定用户发送自定义数据。

返回值：0 表示成功，否则为出错代码

参数：

dwUserId	目标用户 ID
lpBuf	所发送缓冲区地址
dwLen	所发送缓冲区长度，不能大于 1024 个字节

备注：

当用户登录成功之后，无论该用户是否在房间，业务层服务器均可调用该方法向指定的客户端发送数据，数据内容由上层应用自定义。

客户端程序收到服务器所发送的缓冲区数据后，将会触发客户端的回调函数“BRAC_SDKFilterData_CallBack”。

3.22. 向指定房间的所有用户发送数据

DWORD BRAS_SendBufToRoom(DWORD dwRoomId, LPCTSTR lpBuf, DWORD dwLen);

功能：向指定房间的所有用户发送数据。

返回值：0 表示成功，否则为出错代码

参数：

dwRoomId	目标房间 ID，为-1 表示向系统中的所有用户广播
lpBuf	所发送缓冲区地址
dwLen	所发送缓冲区长度，不能大于 1024 个字节

备注：

业务层服务器可使用该方法向指定房间的所有用户广播数据，也可向系统中所有用户广播数据，数据内容由上层应用自定义。

客户端程序收到服务器所发送的缓冲区数据后，将会触发客户端的回调函数“BRAC_SDKFilterData_CallBack”。

3.23. 透明通道传送缓冲区

DWORD BRAS_TransBuffer(DWORD dwUserid, LPBYTE lpBuf, DWORD dwLen);

功能：透明通道传送缓冲区，向指定的用户传输缓冲区

返回值：0 表示成功，否则为出错代码

参数：

dwUserid： 目标用户编号，不能为-1

lpBuf： 缓冲区

dwLen： 缓冲区的大小

备注：

可以利用该方法实现自定义功能，缓冲区采用透明传输，效率与 BRAS_SendBufToRoom、BRAS_SendBufToUser 两个方法相同，只是触发客户端的回调函数不同。

该方法为 V4.1 版本新增 API 接口，主要是为了兼容客户端 API 接口而设计。

该方法将会触发接收到数据的客户端的 BRAC_TransBuffer_CallBack 回调函数。

该方法不需要用户进入房间，即可以在服务器内部的任意用户之间传输数据。

3.24. 透明通道传送缓冲区扩展

DWORD BRAS_TransBufferEx (DWORD dwUserid, LPBYTE lpBuf, DWORD dwLen, DWORD wParam, DWORD lParam, DWORD dwFlags , DWORD& dwTaskId);

功能：透明通道传送缓冲区，传输更大容量的缓冲区数据，而且可以附带参数

返回值：0 表示成功，否则为出错代码

参数：

dwUserid： 目标用户编号，只针对某一个用户，不能为-1（所有人）

lpBuf： 缓冲区，≤1024KB（1MB），内部会自动分包处理

dwLen： 缓冲区的大小

wParam： 附带参数，由上层应用自定义

lParam： 附带参数 2，由上层应用自定义

dwFlags： 特殊功能标志，当对该缓冲区有特殊要求时，可通过使用相关的功能标志，通知 SDK 进行特殊的处理，默认为 0，SDK 将自动根据网络状态选择合适的传输途径（TCP、UDP or P2P）

dwTaskId 与该缓冲区对应的任务 ID（只有任务添加成功后，该 ID 才会自动生成，可利用该 ID 查询该任务的传输进度）

备注：

该方法与“BRAS_TransBuffer”功能相同，都是传输上层应用自定义（透明通道）数据，区别在于该方法通过设置相应的功能标识，如可选择采用 UDP 通道传输，但是只针对指定的用户传输，而“BRAS_TransBuffer”方法则固定采用 TCP 通道传输，且缓冲区大小不能超过 1024 个字节，但可以针对所有用户传输。从应用来看：

（1）、BRAS_TransBuffer 适合数据量小、要求实时传输的缓冲区传递，如控制指令等；

（2）、BRAS_TransBufferEx 适合数据量大、对实时性要求不高的需求；

该方法将会触发接收到数据的客户端的 BRAC_TransBufferEx_CallBack 回调函数。

该方法不需要用户进入房间，即可以在服务器内部的任意用户之间传输数据。

3.25. 传送文件

DWORD BRAS_TransFile(DWORD dwUserid, LPCTSTR lpLocalPathName, DWORD wParam, DWORD lParam, DWORD dwFlags, DWORD& dwTaskId);

功能：传送文件给指定用户

返回值：0 表示任务建立成功（并非文件传输完成），否则为出错代码

参数：

dwUserid： 目标用户编号，只针对某一个用户，不能为-1

lpLocalPathName： 本地文件名，含路径

wParam： 附带参数 1，便于上层应用扩展

lParam： 附带参数 2

dwFlags： 特殊功能标志，参考：“BRAS_TransBufferEx”方法

dwTaskId 与该文件传输对应的任务 ID（只有任务添加成功后，该 ID 才会自动生成，可利用该 ID 查询该任务的传输进度）

备注：

该方法传输效率与“BRAS_TransBufferEx”方法相同，只是在 SDK 内部封装了文件的分组传输功能，实现对上层应用的透明，简化上层应用的开发难度。

客户端收到完整的文件后，将会触发回调函数：BRAC_TransFile_CallBack；

该方法采用 UDP 通道传输，将保证数据传输的可靠性，丢包重传，但不保证接收方接收顺序与发送顺序相同。

该方法不需要用户进入房间，即可以在服务器内部的任意用户之间传输数据。

3.26. 中心端录像控制

DWORD BRAS_StreamRecordCtrlEx(DWORD dwUserId, BOOL bStartRecord, DWORD dwFlags, DWORD dwParam, LPCTSTR lpUserStr, DWORD dwRecordServerId);

功能：向中心录像服务器下达开始（或停止）录像的任务指令

返回值：0 表示指令下达成功（并非录像成功），否则为出错代码

参数：

dwUserid： 目标用户编号，指定需要录像的用户，不能为-1

bStartRecord： 是否开始录像，1 表示开始录像，0 表示结束录像

dwFlags： 标志参数（默认为 0）

dwParam： 用户自定义参数（整形），录像任务结束时，该参数将通过回调函数返回给上层应用

lpUserStr： 用户自定义参数（字符串类型），录像任务结束时，该参数将通过回调函数返回给上层应用

dwRecordServerId 录像服务器 ID（保留，默认为 0）

备注：

AnyChat Server SDK 并不负责音视频录制，而是由专门的中心录像服务器完成，中心录像服务器与核心服务器建立连接，上层应用调用 AnyChat Server SDK 所提供的 API 只是向录像服务器下达开始（或停止）的录像指令，当录像成功之后，将会触发上层应用设定的回调函数。

有关中心录像服务器部署，可参考论坛相关文档：

http://bbs.anychat.cn/forum.php?mod=viewthread&tid=20&extra=page%3D1

3.27. 视频呼叫控制

DWORD BRAS_VideoCallControl(DWORD dwEventType, DWORD dwUserId, DWORD dwErrorCode, DWORD dwFlags, DWORD dwParam, LPCTSTR lpUserStr);

功能：控制（干预）客户端之间的视频呼叫业务。

返回值：0 表示指令执行成功，否则为出错代码

参数：

dwEventType	事件类型，参考“视频呼叫业务”章节相关定义；
dwUserId	控制的目标用户 ID；
dwErrorCode	出错代码；
dwFlags	呼叫标志，默认为 0；
dwParam	附带参数（整型），用户自定义；
lpUserStr	附带参数（字符串），用户自定义；

备注：

目前仅支持挂断（结束）呼叫会话事件类型（BRAC_VIDEOCALL_EVENT_FINISH），即可以使用该接口结束正在进行中的视频通话。

控制的目标用户 ID 为呼叫发起方（或被呼叫方）用户 ID，即通话双方的任意一方用户 ID 均可。

有关视频呼叫业务可参考“视频呼叫业务”章节或在线论坛技术文档：

http://bbs.anychat.cn/forum.php?mod=viewthread&tid=150&extra=page%3D1

3.28. 设置用户的详细信息

DWORD BRAS_SetUserInfo(DWORD dwUserId, DWORD dwInfoId, LPCTSTR lpInfoValue, DWORD dwFlags);

功能：实现客户端用户好友功能，设置用户的详细信息。

返回值：0 表示指令执行成功，否则为出错代码

参数：

dwUserId	设置的用户 ID；
dwInfoId	信息 ID，业务层自定义；
lpInfoValue	参数值，字符串，业务层自定义；
dwFlags	标志参数，默认为 0；

备注：

业务服务器调用该参数设置用户信息后，若将该信息同步给客户端，则客户端可以调用 API：BRAC_GetUserInfo 来获取用户的详细信息。

关于用户信息这一部分，对于 AnyChat 来说是透明的，业务服务器设置了什么样的信息，客户端便可以获取到什么样的信息，AnyChat 只是提供了一个信息传输的中间通道，业务层可以自由扩展。

3.29. 获取用户的详细信息

DWORD BRAS_GetUserInfo(DWORD dwUserId, DWORD dwInfoId, TCHAR* lpInfoValue, DWORD dwSize);

功能：获取用户的详细信息。

返回值：0 表示指令执行成功，否则为出错代码

参数：

dwUserId	用户 ID；
dwInfoId	信息 ID，业务层自定义；
lpInfoValue	保存获取结果的缓冲区；
dwSize	缓冲区大小；

备注：

可以通过该 API 获取已经设置（BRAS_SetUserInfo）的用户详细信息。

3.30. 用户信息控制

DWORD BRAS_UserInfoControl(DWORD dwUserId, DWORD dwCtrlCode, DWORD wParam, DWORD lParam, LPCTSTR lpStrValue);

功能：对用户信息进行控制。

返回值：0 表示成功，否则为出错代码

参数：

dwUserId　　用户 ID；

dwCtrlCode　　控制代码，业务层自定义，其中<100 的值为系统保留，业务层使用时，其值必须>100

wParam　　附带参数，业务层自定义；

lParam　　附带参数，业务层自定义；

lpStrValue　　附带参数（字符串类型），业务层自定义，可为 NULL；

备注：

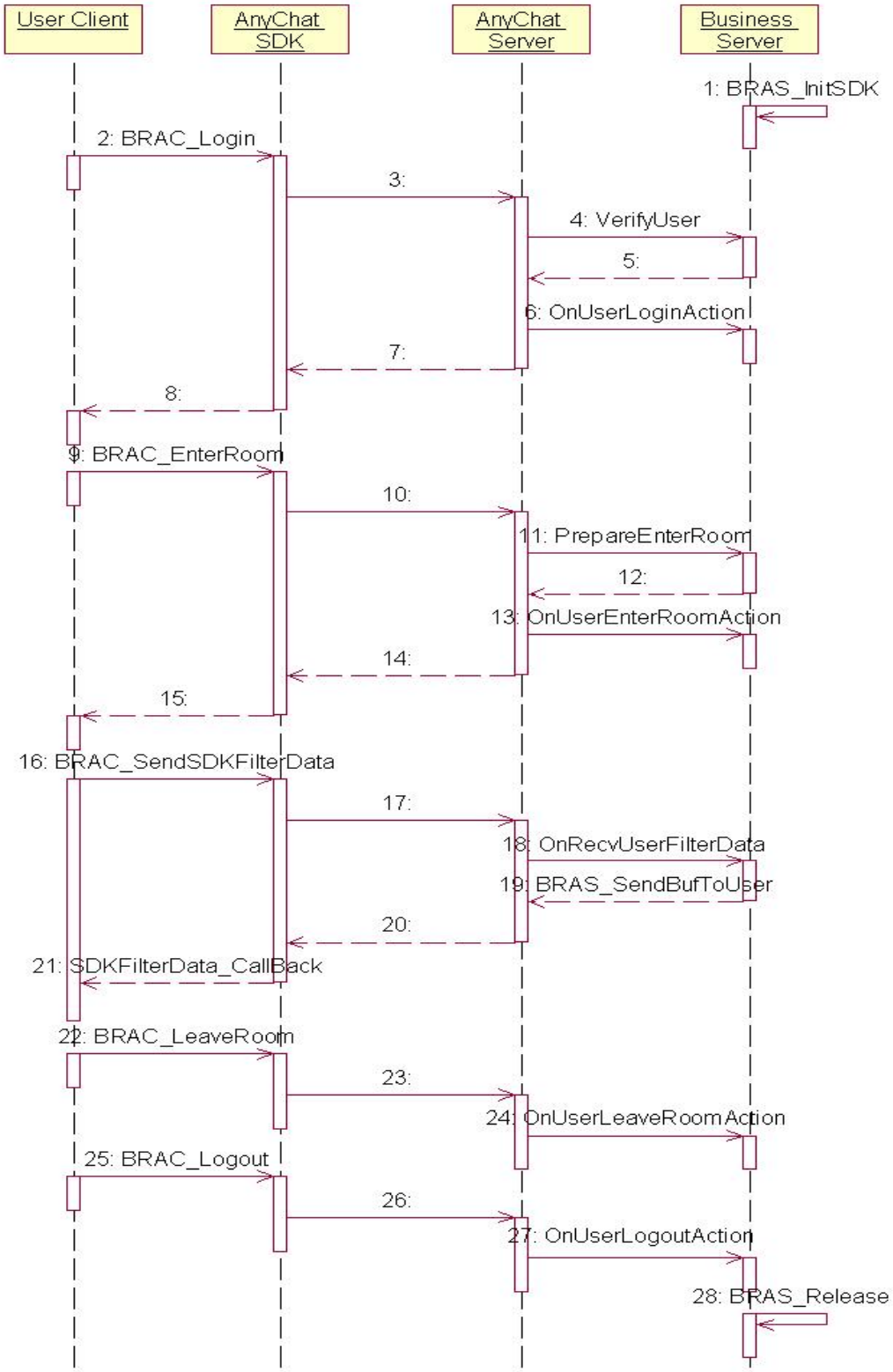
目前已定义的控制代码如下：

```
#define BRAS_USERINFO_CTRLCODE_KICKOUT          1    ///< 将指定用户从系统中踢掉
#define BRAS_USERINFO_CTRLCODE_SYNCDATA         2    ///< 将指定用户的数据同步给客户端

#define BRAS_USERINFO_CTRLCODE_ADDGROUP         20   ///< 添加用户分组，wParam为分组Id，
lpStrValue为分组名称
#define BRAS_USERINFO_CTRLCODE_DELGROUP         21   ///< 删除用户分组，wParam为分组Id
#define BRAS_USERINFO_CTRLCODE_ADDFRIEND        22   ///< 添加用户好友，wParam为好友Id
#define BRAS_USERINFO_CTRLCODE_DELFRIEND        23   ///< 删除用户好友，wParam为好友Id
#define BRAS_USERINFO_CTRLCODE_SETGROUPRELATION     24   ///< 设置好友与分组的关联关系，wParam
为分组Id，lParam为好友Id，表示好友属于某个分组
#define BRAS_USERINFO_CTRLCODE_APPDEFINE        100  ///< 应用层自定义起始指令
```

有关通过用户信息控制来实现用户好友功能，可参考业务服务器示例工程（AnyChatCallCenter），提供了 Java、C#等多种语言的实现方案。

4. 工作流程图


User Client
AnyChat SDK
AnyChat Server
Business Server
1: BRAS_InitSDK
2: BRAC_Login
3:
4: VerifyUser
5:
6: OnUserLoginAction
7:
8:
9: BRAC_EnterRoom
10:
11: PrepareEnterRoom
12:
13: OnUserEnterRoomAction
14:
15:
16: BRAC_SendSDKFilterData
17:
18: OnRecvUserFilterData
19: BRAS_SendBufToUser
20:
21: SDKFilterData_CallBack
22: BRAC_LeaveRoom
23:
24: OnUserLeaveRoomAction
25: BRAC_Logout
26:
27: OnUserLogoutAction
28: BRAS_Release

5.扩展数据传输通道

AnyChat 具有缓冲区及文件传输功能，可以实现客户端与客户端之间的缓冲区和文件传输，也可以实现客户端与服务器之间的缓冲区和文件传输功能（V4.1 版本新增功能）。

AnyChat 客户端之间的缓冲区及文件传输通道有：

1、透明通道 API 接口：BRAC_TransBuffer；

2、透明通道扩展 API 接口：BRAC_TransBufferEx；

3、文件传输 API 接口：BRAC_TransFile；

AnyChat 客户端与服务器之间的缓冲区及文件传输通道有：

1、SDK Filter Data API 接口：BRAC_SendSDKFilterData、BRAS_SendBufToUser、BRAS_SendBufToRoom；

2、透明通道 API 接口：BRAC_TransBuffer、BRAS_TransBuffer；

3、透明通道扩展 API 接口：BRAC_TransBufferEx、BRAS_TransBufferEx；

4、文件传输 API 接口：BRAC_TransFile、BRAS_TransFile；

5.1. 客户端之间使用透明通道传输数据

A、B 两用户登录服务器成功之后，便可使用透明通道 API 接口传输数据，当 A 使用 API 接口：BRAC_TransBuffer 向 B 发送数据时，B 收到数据后，将触发 B 的回调函数：透明通道数据回调函数（参考：BRAC_SetTransBufferCallBack）。

内核使用 TCP 通道传输，保证接收方收到的顺序与发送的顺序相同，且保证数据可达，不会丢包；

5.2. 客户端之间使用透明通道扩展传输数据

A、B 两用户登录服务器成功之后，便可使用透明通道扩展 API 接口传输数

据，当 A 使用 API 接口：BRAC_TransBufferEx 向 B 发送数据时，B 收到数据后，将触发 B 的回调函数：透明通道数据扩展回调函数（参考：BRAC_SetTransBufferExCallBack）。

内核使用 UDP 通道传输，不保证接收方收到的顺序与发送顺序相同，但是保证数据可达，丢包自动重传；

5.3. 客户端之间传输文件

A、B 两用户登录服务器成功之后，便可使用文件传输 API 接口传送文件，当 A 使用 API 接口：BRAC_TransFile 向 B 发送文件时，B 收到文件后，将触发 B 的回调函数：文件传输回调函数（参考：BRAC_SetTransFileCallBack）

内核使用 UDP 通道传输，不保证接收方收到的顺序与发送顺序相同，但是保证数据可达，丢包自动重传；

5.4. 客户端与服务器之间传输缓冲区数据（SDK Filter Data接口）

客户端 A 登录服务器成功之后，便可使用 SDK Filter Data 接口传输数据

1、当客户端使用 API 接口：BRAC_SendSDKFilterData 向服务器发送数据时，将触发服务器的回调函数：BRAS_OnRecvUserFilterData_CallBack；

2、当服务器使用 API 接口：BRAS_SendBufToUser、BRAS_SendBufToRoom 向客户端发送数据时，将触发客户端的回调函数：SDK Filter 通信数据回调函数（参考：BRAC_SetSDKFilterDataCallBack）

内核使用 TCP 通道传输，保证接收方收到的顺序与发送的顺序相同，且保证数据可达，不会丢包；

5.5. 客户端与服务器之间传输缓冲区数据（透明通道接口）

客户端 A 登录服务器成功之后，便可使用透明通道接口与服务器传输数据

1、当客户端使用 API 接口：BRAC_TransBuffer 向服务器发送数据时，将触发服务器的回调函数：BRAS_OnTransBuffer_CallBack；

2、当服务器使用 API 接口：BRAS_TransBuffer 向客户端发送数据时，将触发客户端的回调函数：透明通道数据回调函数；

内核使用 TCP 通道传输，保证接收方收到的顺序与发送的顺序相同，且保证数据可达，不会丢包；

（注：SDK Filter Data 接口和透明通道接口效率相同，透明通道接口主要是为了兼容客户端 API 而设计的，为 V4.1 版本服务器新增接口，需要注意的是两者所对应的回调函数是不同的）

5.6. 客户端与服务器之间使用扩展缓冲区通道传输数据

客户端 A 登录服务器成功之后，便可使用扩展缓冲区通道与服务器传输数据

1、当客户端使用 API 接口：BRAC_TransBufferEx 向服务器发送数据时，将触发服务器的回调函数：BRAS_OnTransBufferEx_CallBack；

2、当服务器使用 API 接口：BRAS_TransBufferEx 向客户端发送数据时，客户端收到数据后，将触发客户端的回调函数：透明通道数据扩展回调函数（参考：BRAC_SetTransBufferExCallBack）。

内核使用 UDP 通道传输，不保证接收方收到的顺序与发送顺序相同，但是保证数据可达，丢包自动重传；

5.7. 客户端与服务器之间传输文件

客户端 A 登录服务器成功之后，便可使用文件传输接口与服务器传输文件

1、当客户端 A 使用 API 接口：BRAC_TransFile 向服务器发送文件时，服务器收到文件后，将触发服务器端的回调函数：BRAS_OnTransFile_CallBack；

2、当服务器使用 API 接口：BRAS_TransFile 向客户端发送文件时，客户端收到文件数据后，将触发客户端的回调函数：文件传输回调函数（参考：BRAC_SetTransFileCallBack）；

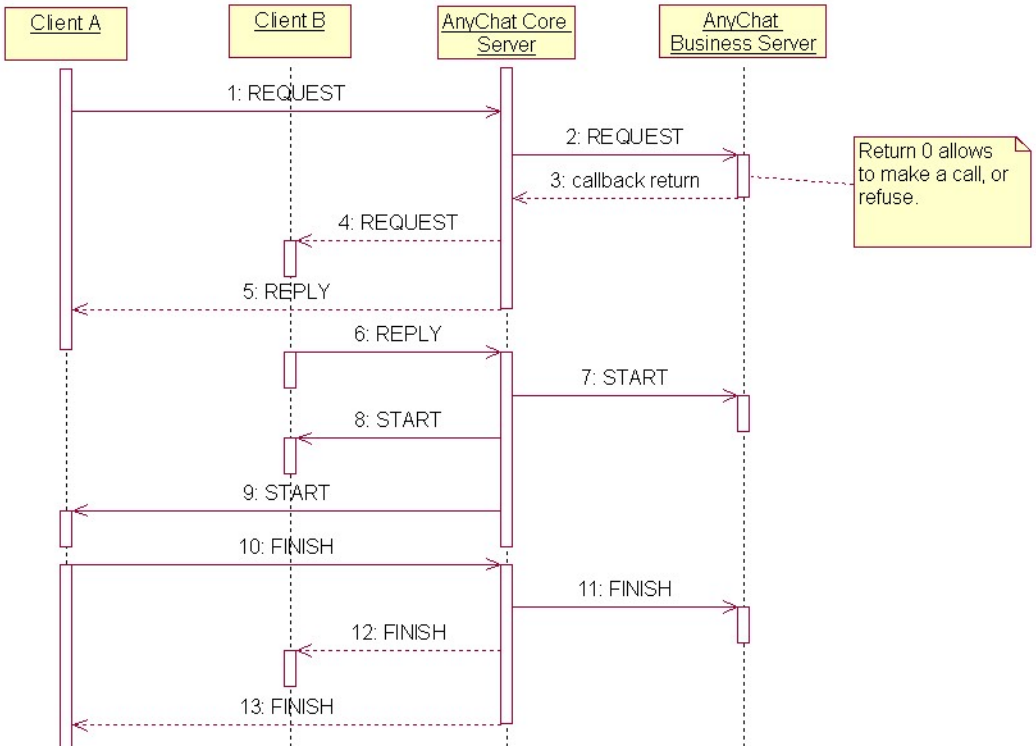
内核使用 UDP 通道传输，不保证接收方收到的顺序与发送顺序相同，但是保证数据可达，丢包自动重传；

6.视频呼叫业务

视频呼叫业务逻辑主要实现两个终端（PC、手机、Pad 等）之间的通话请求流程控制，包括请求（Request）、回复（Reply）、开始（Start）以及结束（Finish）等过程，可以形象理解为打电话的流程：拨号、等待、通话、挂断。

AnyChat SDK V4.8 之后的版本才能支持视频呼叫业务。

6.1. 业务逻辑流程图



6.2. API接口及常量定义

客户端相关 API 接口及事件回调：

```
// 视频通话消息通知回调函数定义
typedef void (CALLBACK * BRAC_VideoCallEvent_CallBack)(DWORD dwEventType, DWORD dwUserId, DWORD dwErrorCode, DWORD dwFlags, DWORD dwParam, LPCTSTR lpUserStr, LPVOID lpUserValue);
// 设置视频通话消息通知回调函数
BRAC_API DWORD BRAC_SetVideoCallEventCallBack(BRAC_VideoCallEvent_CallBack lpFunction, LPVOID lpUserValue);
// 视频呼叫事件控制（请求、回复、挂断等）
BRAC_API DWORD BRAC_VideoCallControl(DWORD dwEventType, DWORD dwUserId, DWORD dwErrorCode, DWORD
```

```
dwFlags, DWORD dwParam, LPCTSTR lpUserStr);
```

服务器相关 API 接口及事件回调：

```
// 视频通话消息通知回调函数定义
typedef DWORD (CALLBACK * BRAS_OnVideoCallEvent_CallBack)(DWORD dwEventType, DWORD dwSrcUserId, DWORD dwTarUserId, DWORD dwErrorCode, DWORD dwFlags, DWORD dwParam, LPCTSTR lpUserStr, LPVOID lpUserValue);
// 设置视频通话消息通知回调函数
BRAS_API DWORD BRAS_SetOnVideoCallEventCallBack(BRAS_OnVideoCallEvent_CallBack lpFunction, LPVOID lpUserValue=NULL);
// 视频呼叫事件控制（请求、回复、挂断等）
BRAS_API DWORD BRAS_VideoCallControl(DWORD dwEventType, DWORD dwUserId, DWORD dwErrorCode, DWORD dwFlags, DWORD dwParam, LPCTSTR lpUserStr);
```

常量定义

```
/ 视频呼叫事件类型定义（API：BRAS_VideoCallControl 传入参数、OnVideoCallEvent回调参数）
#define BRAS_VIDEOCALL_EVENT_REQUEST    1       ///< 呼叫请求
#define BRAS_VIDEOCALL_EVENT_REPLY      2       ///< 呼叫请求回复
#define BRAS_VIDEOCALL_EVENT_START      3       ///< 视频呼叫会话开始事件
#define BRAS_VIDEOCALL_EVENT_FINISH     4       ///< 挂断（结束）呼叫会话
```

6.3. 相关说明

1、客户端 API（BRAC_VideoCallControl）和回调函数（BRAC_VideoCallEvent_CallBack）中的 dwUserId 均为对方（被呼叫方）的用户 ID；

2、被呼叫方拒绝通话时，发送回复（Reply）指令，dwErrorCode=100104；

3、被呼叫方同意通话时，发送回复（Reply）指令，dwErrorCode=0，然后服务器会向双方发送通话开始（Start）指令，dwParam=RoomId，房间号由核心服务器自动分配；

4、结束通话时，任何一方（包括业务服务器）均可以发送结束（Finish）指令，然后服务器会向双方发送通话结束（Finish）指令；

5、业务服务器可干预呼叫流程：在 BRAS_OnVideoCallEvent_CallBack 收到

呼叫请求指令后，返回 0 表示允许呼叫，否则为出错代码，不允许呼叫；在会话过程中可以发送结束（Finish）指令，强制挂断指定用户的通话；

6、API 接口中的 dwParam（整型）、lpUserStr（字符串）均为用户自定义用途；

7、一个用户同时只能发起一路呼叫请求，也同时只能被一个用户呼叫；

8、视频呼叫业务流程可以脱离业务服务器，由核心服务器独立支撑，可以不需要在服务器端进行二次开发。

9、更多关于视频呼叫业务流程信息可参考在线论坛技术文档：http://bbs.anychat.cn/forum.php?mod=viewthread&tid=150&extra=page%3D1

7. SDK使用与参数配置

“AnyChat Server SDK”的正常工作需要服务器程序的“SDK Filter Plus”插件支持，两者之间采用 IPC 接口进行通信，从而实现业务层服务器与 AnyChat Core Server 运行在两个不同的进程中。

为了能让“AnyChat Server SDK”能正常工作，必须将具备IPC通信功能的“SDK Filter Plus”插件配置到服务器的配置文件中，让服务器启动时加载所配置的“SDK Filter Plus”插件，这样便可以让“AnyChat Server SDK”与“AnyChat Core Server”之间建立连接。

在服务器的配置文件 AnyChatCoreServer.ini 中，默认有如下两项参数配置项：

```
[Function Settings]
CloseFrontLink=1
SDKFilterPlus=
AutoAddRoom=1
```

其中“SDKFilterPlus”配置项需要配置“AnyChat Server SDK”所对应的“SDK Filter Plus”插件（在 Server SDK 目录下，名称为“IPCFilterPlus.dll”）。修改后的配置文件相关内容如下：

```
[Function Settings]
CloseFrontLink=1
SDKFilterPlus=IPCFilterPlus.dll
AutoAddRoom=1
```

同时需要将插件的 DLL 文件（在 Server SDK 目录下，名称为“IPCFilterPlus.dll”）拷贝到服务器程序所在的目录，保存配置文件后，最后重启服务器程序，通过服务器的日志文件“AnyChatCoreServer.log”便可判断 SDK Filter Plus 插件是否被服务器加载成功。

8. 常用业务处理逻辑

不同的应用有不同的业务处理逻辑，下面列举了一些常见应用并结合AnyChat平台的相关特性来说明这些应用的业务逻辑的实现方案供参考，实际中的一些应用大多会比较复杂，希望通过常用业务处理逻辑的表述能起到抛砖引玉的作用。

8.1. IM中好友列表的实现

由于AnyChat Platform Core SDK只能在进入房间后，才能获取房间用户列表，而不能实现登录成功之后获取好友列表的功能，这项功能需要通过业务层扩展来实现，具体实现方法如下：

1、客户端登录成功后，会触发业务层服务器的回调函数：“OnUserLoginAction_CallBack”，业务服务器可以在该回调函数里面将该用户的好友信息（ID、昵称、在线状态）通过API接口“BRAS_SendBufToUser”发送给该用户，同时将该用户的上线信息发送给该用户的所有在线好友，即“当自己登录服务器成功后，服务器将好友信息发给我，同时将自己的信息发送给其它的好友”；

2、客户端收到服务器发送的好友信息后，便可显示好友列表及其在线状态；

3、其它客户端收到好友的上线、下线消息后，便可更新好友的在线状态；

4、当客户端程序关闭，或是网络掉线后，会触发业务层服务器的回调函数：“OnUserLogoutAction_CallBack”，业务服务器可以在该回调函数中向该客户端的所有好友发送“好友下线”的消息；

8.2. IM中点对点聊天的实现

由于AnyChat Platform Core SDK只能在房间里面才能语音、视频以及文字的交互，所以如果想在AnyChat实现的IM中完成语音、视频的点对点聊天，则需要通过业务层扩展来实现，具体实现方法如下：

1、客户端A想与客户端B进行语音视频通话，则A通过透明通道函数（BRAC_TransBuffer）向B发送一个语音视频聊天的请求，当B同意后，使

用接口（BRAC_SendSDKFilterData）通知业务层服务器，请求分配一个点对点聊天的房间；

2、 业务层服务器收到 A、B 的点对点聊天请求后，分配一个房间号，使用接口（BRAS_SendBufToUser）通知到 A、B，并在通知的参数中附带房间号；

3、 A、B 收到业务层服务器的通知后，同时进入由服务器分配的房间，打开自己的语音、视频设备，并自动请求对方的语音、视频数据，这样便实现了 A、B 两个用户在同一个房间中进行语音、视频聊天的功能，而进入房间这个过程对用户来说完全是透明的；

4、 当 A、B 有一方关闭聊天界面（离开房间）时，另外一方也自动离开房间；

5、 登录系统成功之后，没有进入房间时，如果 A 希望向 B 用户发送文字消息，上层应用可自定义一条指令，通过接口“BRAC_TransBuffer”发送给对方，即点对点的文字聊天，可以不用进入房间，通过透明通道即可实现；

8.3. 聊天室中公麦的实现

在AnyChat Platform Core SDK的平台中，没有用户拿麦的概念，只有打开本地设备（音频设备、视频设备），以及请求远程用户的流数据（语音流、视频流），当某用户打开了本地设备后，其它用户请求该用户的数据时，便能收到该用户的数据。（例如在房间中，共有A、B、C三个用户，均打开了本地设备，其中A没有请求任何用户的语音视频，B请求了A的语音、视频，同时请求了C的语音视频，C只请求了A的语音视频，则B、C两个用户可以听到A的声音，看到A的视频，B用户可以看到C的视频，听到C的声音，而A用户不会收到其它用户的声音和视频数据）

在上述 AnyChat 特性的基础上，房间中的公麦便可通过业务服务器来实现，具体实现方式如下：

1、在业务层服务器上为对应的房间配置合适的公麦数（如 1、2、3、4 个等，通常是 2 个，具体数量与应用相关），默认没有用户上麦；

2、当有用户请求上麦时，业务层服务器判断是否有空闲的公麦，如果有，则为该用户分配一个公麦序号，并向房间广播某用户上麦的消息，并在消息中附带公麦序号参数；

3、当房间内的用户收到业务层服务器发送的某用户上麦的指令后，便请求该用户的语音、视频数据，并将用户的视频显示到指定的区域，这样房间内的所有人均可以听到该用户的声音了，也可以看到该用户的视频了（默认该用户已打开本地的语音、视频设备）；

4、当有新的用户进入房间时，在业务层服务器的回调函数“OnUserEnterRoomAction”中，业务层服务器将当前公麦的状态（谁在上麦）同步给新用户，新用户收到业务层的公麦状态后，如果有用户上麦，则请求该用户的语音、视频数据，这边便实现了新用户一进房间，就可以看到公麦用户的视频，听到公麦用户的语音。

5、当用户下麦（包括用户主动下麦、上麦超时业务服务器切断）时，业务服务器向房间内的所有用户广播下麦消息。

8.4. 聊天室中麦序的实现

基于“聊天室中公麦的实现”中所描述的 AnyChat 相关特性，AnyChat 平台中没有“麦序”，或是“排麦”的概念，这需要业务层服务器来实现，具体实现逻辑如下：

1、业务层服务器为每一个房间保存一个麦序列表，当用户进行“排麦”操作时，如果“公麦”空闲，则直接让用户上麦，如果公麦已满，则进入麦序列表，并向所有客户端广播麦序列表，客户端便可显示麦序列表；

2、当新用户进入房间时，在在业务层服务器的回调函数“OnUserEnterRoomAction”中，业务层服务器将当前麦序列表同步给新进用户；

3、当公麦用户主动放麦，或是上麦超时（业务层服务器可控制每个用户上麦的时间），业务层服务器让麦序列表中排在最前面的用户上麦，向房间广播用户上麦消息，重新调整麦序列表，并向房间所有用户广播新的麦序列表；

4、当用户离开房间时，而且该用户在麦序列表中，业务层服务器需要更新麦序列表，并向房间内的所用户进行麦序列表的同步。

9. 技术支持

在使用 “AnyChat Platform Core SDK”进行开发的过程中遇到异常情况，可开启“AnyChat Server SDK”的调试模式，让“AnyChat Server SDK”输出调试信息到“AnyChatServerSDK.log”文件中。

开启或关闭“AnyChat Server SDK”调试模式的方法是修改配置文件“AnyChatServerSDK.ini”，其中的配置项“DebugMode=0”（默认）表示关闭调试信息输出，“DebugMode=1”表示开启调试信息输出。

在您使用 AnyChat SDK 的过程中，遇到任何困难，请与我们联系，我们将热忱为您提供帮助。

您可以通过如下方式与我们取得联系：

1、在线论坛：http://bbs.anychat.cn/

2、知识中心：http://www.anychat.cn/faq/

3、官方网站：http://www.anychat.cn

4、电子邮件：service@bairuitech.com

5、24 小时客服电话：+86 （020） 85276986、38109065、38103410