**1.机制（会话中）**

1）web-client端通过接口A通知远端变化视频参数

2）mcu端根据接口B的传入参数修改视频会话质量参数

**3.策略**  
1）web-client页面上添加功能按钮，用户手动发起质量调整请求，调用**接口A**；

2）接口A通过webrtc的message通道发送降低/提升视频质量（模糊/精确）的**消息**；  
3）mcu端处理接收的消息，如果为视频调整请求，则调用**接口B**

4）mcu端可以通过设置libjingle-callwebrtc-0-video的bitrate size属性值，然后设置needLoadSets属性来主动调节

发送的视频质量(后期可能这些操作步骤会提取为**接口C**)

注：自动调整的策略涉及到网络层丢包、应用层解码等能力不够的发现，暂时先不考虑，先做手动方式的策略

=====================================================

**1、接口A定义：**

/\*

参数:

value 字符串,取值参见下面[消息内容定义]的value说明

type  字符串,取值参见下面[消息内容定义]的type说明

说明:

用户可以短时间内多次请求controlVideoQp接口,该接口会延迟[delaySendQpMsg]毫秒后统计、处理用户的请求，

生成webrtc-MSG发送给对端

\*/

function controlVideoQp(value, type)

**涉及代码：**

**index.html**

添加<div id="videoQp-control">块,提供给用户发起请求

**manual/peerconnection.js**

var videoQpTimer = null;  
var videoQpMsgSize = 0;  
var videoQpMsgs = [];  
var delaySendQpMsg = 1500;//in ms  
function controlVideoQp(value, type)  
function callQqChanged()  
function sendVideoQpMsg()

**消息内容定义：**

方法method: video-qp ...

方向dir :rx tx(相对于消息发送者,目前只支持rx)

类型type :bitrate size both...

变量value :

(1)up down  (size step:320x240 640x480 1280x720 1920x1080) (bitrate step:50 192 384 768 1536 2000)

(2)width[x]heigh(just for size)

(3)50~2000(just for bitrate)

结束标志end:yes/no  表示是否为参数设置的最后一个消息

例：

POST /message?peer\_id=xx&to=xx HTTP/1.0  
Content-Length: xx  
Content-Type: text/plain  
  
{  
  
"method" : "video-qp",

"type" : "bitrate",

"dir" : "rx",

"value" : "down"

"end" : "yes"  
}

**接口B定义:**

**/\***

**参数:**

info  消息内容,详细参见CUS\_MSG::VIE\_QP\_INFO定义

说明：

该接口将回调给VMX层来处理,当消息中end值为yes时,才会真正改变video流质量

**\*/**

void UserHandler::onVideoQpMessage(CUS\_MSG::VIE\_QP\_INFO \*info);

**涉及代码：**

添加 UserHandler::onVideoQpMessage  
添加 UserHandler::ResetStreams()  
添加 UserHandler::UserHandlerEvents::Evt\_ResetLocalStream  
修改 UserHandler::OnEvent::Evt\_ResetLocalStream  
添加 struct CUSTOM\_MSG,EXCMD::CMD\_ON\_RECVCUSTOMMESSAGE  
  
修改 Conductor::OnMessageFromPeer  mark with //custom message  
添加 Conductor::RemoveStreams()  
  
修改 WebrtcClient.cpp:vmxExCmdCb  
添加 WebrtcClient.cpp:若果变量、函数 mark with //tuyj video-qp  
添加 WebrtcClient::getNodeProperty,setNodeProperty  
修改 LibjingleAdapterNode::onSetProperty,onGetProperty  
添加 LibjingleAdapterNode属性PROPERTY\_WEBRTC\_VIDEO\_LOADSETS

**TODO:**

stream 什么时候析构

stream使用到的audio-device video-capture什么时候析构

有没有内存泄露问题？

从理论上看应该没有问题，暂没测试验证，以后再说