集成方案核心文档

本文档主要说明如何使用 ISAPI 协议集成网络摄像机。ISAPI 协议定义了访问摄像机各种资源的通用接口，文档将逐步介绍使用 ISAPI 协议访问，控制网络摄像机的方法。

本集成方案一共包括 5 个部分：《ISAPI 集成方案核心文档》，《RTSP 协议开发接口说明》，《ISAPI 事件功能开发文档》，《ISAPI PTZ 功能开发文档》，《ISAPI ISAPI 图像显示功能开发》。

《集成方案核心文档》是集成方案的基础部分，该文档介绍协议的原理，操作机制，以及基础功能开发操作，比如系统维护，视频参数控制，网络功能开发，报警输入报警输出控制。该文档是《ISAPI 事件功能开发文档》，《ISAPI PTZ 功能开发文档》，《ISAPI 图像显示功能开发》的基础。

《RTSP 协议开发接口说明》主要说明如何使用 RTSP 协议完成设备取流操作会话。

《ISAPI 事件功能开发文档》是关于事件功能开发的方案说明文档，事件主要涵盖移动侦测，报警输出，遮蔽告警等事件内容。《图像显示功能开发》是关于前端图像参数和图像显示操作的集成方案文档，该部分功能主要包括设置字符叠加显示，前端参数控制，比如亮度，色度，对比度，饱和度，日夜切换等功能。

《ISAPI PTZ 功能开发文档》是关于云台控制功能的集成方案文档，主要介绍云台指令的使用。

目录

1 ISAPI 协议简介

1.1 ISAPI 协议概述

ISAPI 协议是基于 HTTP 协议的一套 CGI 接口，支持 HTTP 协议中的 GET,PUT,POST,DELETE 方法，这四种方法的一般含义如下：

GET 方法： 主要应用于获取资源，该操作方式不改变资源存在的状态，请求合法的情况下将按照协议格式返回资源信息。

PUT 方法： 更新资源，重新设置资源，该方法会改变资源，设备会返回操作的结果，使用 ResponseStatus 页面表明此次操作的具体结果信息。

POST 方法：主要用于创建资源，创建的资源此前不存在，设备将按照 POST 命令携带的参数创建该资源，并返回资源的 ID，创建资源的操作响应与 PUT 命令一样，都是以 ResponseStatus 表明操作的具体结果。除此之外，POST 命令还有可能查询，对于查询条件较为复杂的请求，可以将请求放入 HTTP Body，通过 POST 命令发送。

DELETE 方法：用于删除资源，删除的资源必须已经存在，否则将操作失败，该命令的响应也是以 ResponseStatus 页面表示的。

1.2 HTTP 协议简介

ISAPI 协议是基于 HTTP 协议，因此有必要先对 HTTP 协议有简要的了解。

HTTP 协议是面向字符的协议，协议分为头部（HTTP Head）和实体（HTTP body）部分。

头部中的主要信息为 HTTP 方法（HTTP method），URL，实体长度，实体类型。**头部的内容分行表示，每行的结尾均以换行表示结束（\r\n,或者\n），整个头部结束使用连续两个换行标志表示**。头部中除第一行之外，其他各行一般是对 HTTP 实体的说明。举例如下：

（1）示例 1. 查询设备状态信息

```
GET /ISAPI/System/deviceInfo HTTP/1.1
Host: 172.8.6.155
Authorization: Basic YWRtaW46MTIzNDU=
```

这是一个 HTTP 请求，该请求的方法（HTTP Method）为 GET ， URL 路径为/ISAPI /System/deviceInfo。

头部字段的第二行 Host 指明发往的目的主机的 IP 地址是 172.8.6.155.

头部字段的第三行是认证字段，目前摄像机使用的认证加密方式为 Basic，所以这个请求中的认证方式为 Authorization: Basic,。用户名是 admin,密码是 12345，”YWRtaW46MTIzNDU=”实际上是对“admin:12345”进行 base64 加密的结果。

头部中的每一行都是以”\r\n”作为结束的，**整个头部结束后，会有一个空行**，即只有“\r\n”作为整个头部结束标志。

这条指令实际上是获取设备的信息，IPC 回复如下：

```
HTTP/1.1 200 OK
Date: Fri, 13 Jul 2012 16:38:16 GMT
Server: App-webs/
Connection: close
Content-Length: 693
Content-Type: application/xml
X-Appweb-Seq: 19697

<?xml version="1.0" encoding="UTF-8"?>
<DeviceInfo version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
<deviceName>IP CAMERA</deviceName>
…….
</DeviceInfo>
```

IPC 的回复由 HTTP header 和 HTTP body 组成，HTTP Header 中，第一行由状态码 200 和状态信息 OK 表示此次之行成功，关于 HTTP 状态码可以参考协议附录。

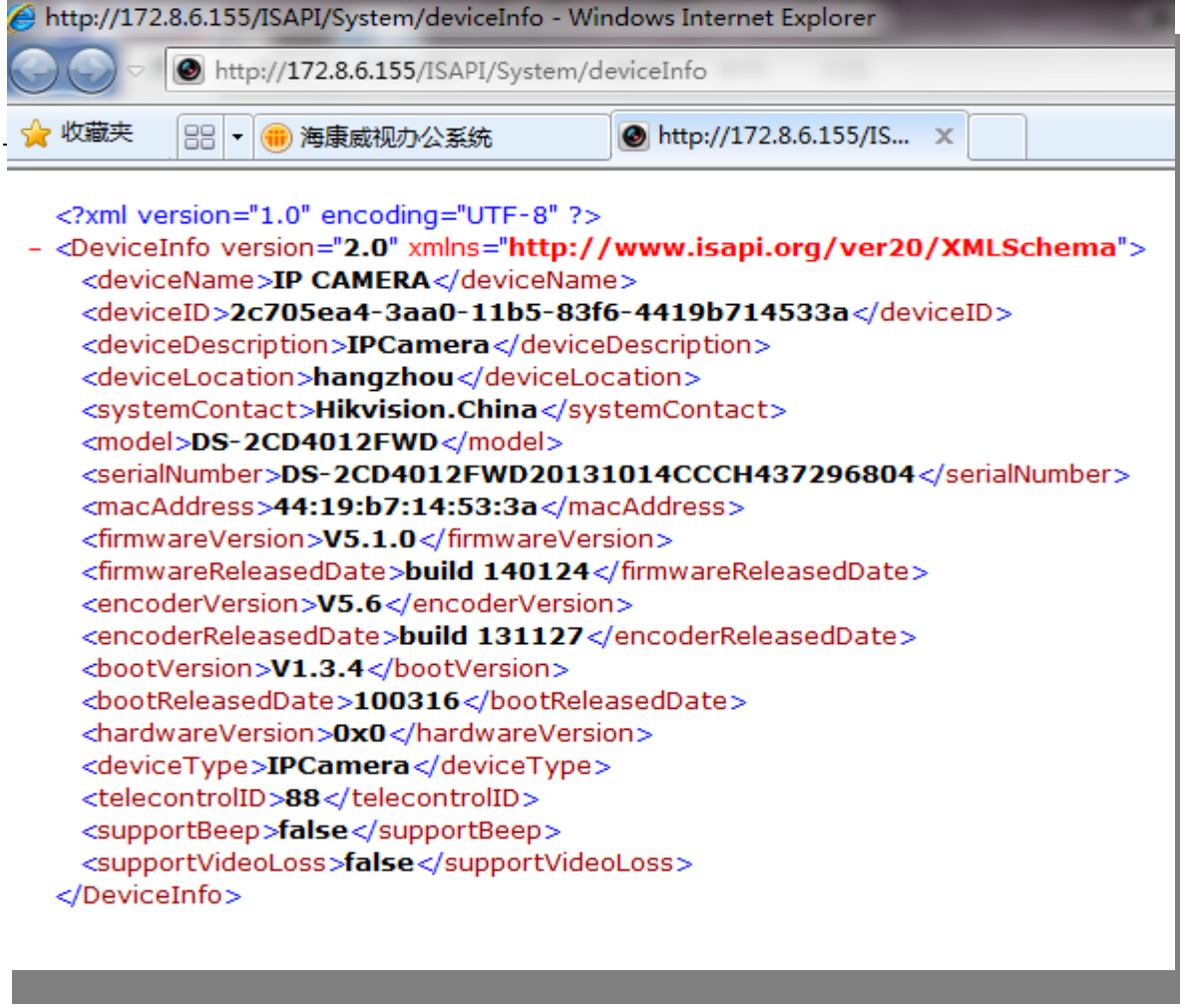
头部第五行 Content-Length 字段指明了 HTTP body 的长度是 693 个字节。

第六第六行 Content-Type 字段指明类 HTTP body 的内容格式是 XML 格式的(Content-Type 常见格式说明请参考 8.1 小节)。

头部结束之后就是 HTTP body，是关于设备信息，采用 XML 格式表示，这里只是给出一个样式，并没有全部列出。

GET 命令主要用于获取设备信息，一般情况下请求中不包含 HTTP body，所以 GET 请求的 HTTP Header 中一般不包含 Content-Length 字段和 Content-Type 字段。

实际上，对于 HTTP GET 命令，一般的浏览器有很好的支持，比如获取设备信息，可以直接在 IE 浏览器中输入如下信息：
http://ipaddress/ISAPI/System/deviceInfo，浏览器可能会询问用户名和密码，输入



正确信息后，会出现上面的界面：

（2）示例 2.设置 IPC 时钟

```
PUT /ISAPI/System/time/localTime HTTP/1.1
Authorization: Basic YWRtaW46MTIzNDU=
Content-Type: text/plain
Host: 172.8.6.155
Content-Length: 19

2012-07-13T17:02:33
```

头部中指示 HTTP body 的格式是普通文本格式，长度为 19 个字节。

头部结束后，HTTP body 是一个关于时间信息的字符串，指示了要设置的时间。

PUT 命令一般用于设置操作，PUT 操作方式一般都会携带 HTTP body 指明操作参数，因此 PUT 请求的 HTTP Header 中一般会有 Content-Type 字段和 Content-Length。

IPC 的返回信息如下：

```
HTTP/1.1 200 OK
Content-Length: 280
```

```
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<ResponseStatus version="2.0"
xmlns="http://www.isapi.org/ver20/XMLSchema">
<requestURL>/ISAPI/System/time/localTime</requestURL>
<statusCode>1</statusCode>
<statusString>OK</statusString>
<subStatusCode>ok</subStatusCode>
</ResponseStatus>
```

回复同样由 HTTP 头部和 HTTP body 组成，头部中的第一行的状态码 200，指示此命令执行成功。

HTTP body 是 ResponseStatus 页面，ResponseStatus 中会给出比较详细的反馈信息，<requestURL>指示了这次回应是针对什么请求的，<statusString>是具体的反馈信息（关于 HTTP code 和 statusCode 请参考协议文档 4.5.5Error Handling）。

1.3 ISAPI 协议文档使用说明

ISAPI 协议将各功能模块称为 Service，Service 进一步细分为 Resource，以以 8.2.4 小节为例，这是一个关于网络摄像机 IP 地址访问的资源，文档中定义具体含义可以参考下图：

/ISAPI/System/Network/interfaces/*ID*/ipAddres	**General Resource**
GET	
Description	It is used to get the ip address of a particular network interface.
Query	None
Inbound Data	None
Success Return	**IPAddress**
PUT	
Description	It is used to update the ip address of a particular network interface.
Query	None
Inbound Data	**IPAddress**
Success Return	**ResponseStatus**
Notes: If <addressingType> is dynamic, fields below it need not be provided. If <addressingType> is dynamic, a DHCP client is used for the device. If <addressingType> is static the device IP address is configured manually and the gateway and DNS fields are optional.	

If <addressingType> refers to APIPA, the device IP address is automatically configured without DHCP. In this case the gateway and DNS fields are optional.

Use of <ipAddress> or <ipv6Address> in fields is dictated by the <ipVersion> field. If <ipVersion> is “v4” the <ipAddress> fields are used; if <ipVersion> is “v6” the <ipv6Address> fields are used. If <ipVersion> is "dual", both <ipAddress> and <ipv6Address> fields may be used.

<subnetMask> notation is “ISAPI.ISAPI.ISAPI.ISAPI”.

<IPV6Address> is “ISAPIx:ISAPIx:ISAPIx:ISAPIx:ISAPIx:ISAPIx:ISAPIx:ISAPIx” using CIDR notation.

顶行声明该资源的 HTTP URL 路径；

文档里面标示了该资源可以使用 HTTP GET 和 HTTP PUT 方法访问，这意味着该资源既可以获取，也可以被修改;

每个 HTTP 方法下的 Description 是对该方法的描述，说明该方法的作用；

每个方法下的“Inbound Data”指明该方法是否携带 HTTP body，以及携带什么内容形式的 HTTP body。比如 GET 方法下的 Inbound Data 注明为“None”，这说明使用 GET 方法访问该资源不需要 HTTP body，只需要构造好 HTTP 头部发送过来即可；而 PUT 方法下的 Inbound Data 注明为“IPAddress”，表示该方法需要有 HTTP body，HTTP body 的内容应当按照协议制定的“IPAddress”XML block 进行构造，小节中对 IPAddress XML block 进行了定义：

IPAddress XML Block

```
<IPAddress version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
<ipVersion>  <!-- req, xs:string, "v4,v6,dual" --></ipVersion>
<addressingType>  <!-- req, xs:string, "static,dynamic,apipa" -->
</addressingType>
<ipAddress> <!-- dep, xs:string -->  </ipAddress>
<subnetMask>  <!-- dep, xs:string, subnet mask for IPv4 address -->
</subnetMask>
<ipv6Address>  <!-- dep, xs:string -->  </ipv6Address>
<bitMask>  <!-- dep, xs:integer, bitmask IPv6 address -->  </bitMask>
<DefaultGateway>  <!-- dep -->
<ipAddress> <!-- dep, xs:string -->  </ipAddress>
<ipv6Address> <!-- dep, xs:string -->  </ipv6Address>
</DefaultGateway>
<PrimaryDNS>  <!-- dep -->
<ipAddress> <!-- dep, xs:string -->  </ipAddress>
<ipv6Address> <!-- dep, xs:string -->  </ipv6Address>
</PrimaryDNS>
<SecondaryDNS>  <!-- dep -->
<ipAddress> <!-- dep, xs:string -->  </ipAddress>
<ipv6Address> <!-- dep, xs:string -->  </ipv6Address>
</SecondaryDNS>
```

```
</IPAddress>
```

XML block 详细定义了每个标签的属性，标志为“req”的是必须要出现的，标志为“opt”的，是可选的，可以出现也可以不出现（标签的属性介绍详见 6.3 Annotation）。如果是 PUT 请求，对于 opt 属性的标签，如果请求中有该标签，该标签的值会被改变，如果没有出现，会保持原值。

2 获取设备能力

在接入设备时，首先需要获得的就是设备能力信息，包括：设备基本信息，设备硬件能力，设备支持的视频通道等信息。

2.1 获取设备基本描述信息

设备基本信息包括设备的类型，设备型号，设备序列号等信息。

命令： GET /ISAPI/System/deviceInfo

示例如下：

```
GET /ISAPI/System/deviceInfo HTTP/1.1
Authorization: Basic YWRtaW46MTIzNDU=
Host:172.8.6.155
Content-Type:text/xml
```

IP 摄像机应答如下：

```
HTTP/1.1 200 OK
Date: Wed, 28 Mar 2012 10:56:50 GMT
Connection: close
Content-Length: 959
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<DeviceInfo version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
<deviceName>IP CAMERA</deviceName>
<deviceID>2c705ea4-3aa0-11b5-83f6-4419b714533a</deviceID>
<deviceDescription>IPCamera</deviceDescription>
<deviceLocation>STD-CGI</deviceLocation>
<systemContact>STD-CGI</systemContact>
<model>DS-2CD4012FWD</model>
<serialNumber>DS-2CD4012FWD20131014CCCH437296804</serialNumber>
```

```
<macAddress>44:19:b7:14:53:3a</macAddress>
<firmwareVersion>V5.1.0</firmwareVersion>
<firmwareReleasedDate>build 140213</firmwareReleasedDate>
<encoderVersion>V5.6</encoderVersion>
<encoderReleasedDate>build 131127</encoderReleasedDate>
<bootVersion>V1.3.4</bootVersion>
<bootReleasedDate>100316</bootReleasedDate>
<hardwareVersion>0x0</hardwareVersion>
<deviceType>IPCamera</deviceType>
<telecontrolID>88</telecontrolID>
<supportBeep>false</supportBeep>
<supportVideoLoss>false</supportVideoLoss>
</DeviceInfo>
```

<deviceNam>是设备的名称，可以被修改，<deviceDescriptio>是对设备的描述，是只读信息，目前，网络摄像机的描述是“IPCamera”；对网络快球的描述是“IPDome”；对 DVR 的描述是“DVRDVS”。<mode>是设备的型号。<supportBeep>表示是否支持声音告警。

2.2 获取设备的硬件信息

设备提供的功能依赖于设备当前的硬件能力，比如对于没有音频通道的设备不能提供复合流，而只能提供视频流。

设备的硬件能力包括：报警输入端口的数量；报警输出端口的数量；音频通道的数量；视频通道的数量；是否支持云台。

2.2.1 获取设备报警输入数量

命令： GET /ISAPI/System/IO/inputs

示例如下：

```
GET /ISAPI/System/IO/inputs HTTP/1.1
Host: 172.8.6.155
Connection: Keep-Alive
Authorization: Basic YWRtaW46MTIzNDU=
```

获取到的信息是一个报警输入列表，如下：

```
HTTP/1.1 200 OK
Date: Sat, 14 Jul 2012 05:42:40 GMT
```

```
Server: App-webs/
Connection: close
Content-Length: 422
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<IOInputPortList xmlns="http://www.isapi.org/ver20/XMLSchema">
<IOInputPort >
<id>1</id>
<triggering>high</triggering>
</IOInputPort>
<IOInputPort >

<id>2</id>
<triggering>high</triggering>
</IOInputPort>
<IOInputPort >
<id>3</id>
<triggering>high</triggering>
</IOInputPort>
<IOInputPort>
<id>4</id>
<triggering>high</triggering>
</IOInputPort>
</IOInputPortList>
```

<IOInputPortList>是一个列表，列表中列举了所有的报警输入端口，每一个报警输入端口都由一个<IOInputPort>表示，在这个列表中共计出现了四个IOInputPort，所以这个设备支持四个报警输入端口，triggering 代表触发报警输入的电平，如果<triggering>为”high”，代表高电平将触发报警输出。

某些设备可能没有报警输入，对于这样的设备，<IOInputPortList>将会是一张空表：

```
<?xml version="1.0" encoding="UTF-8"?>
<IOInputPortList version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
</IOInputPortList>
```

如上所示，<IOInputPortList>中没有任何<IOInputPort>的信息，这是一个没

有报警输入的设备

2.2.2 获取设备报警输出数量

命令：GET /ISAPI/System/IO/outputs

示例如下：

```
GET /ISAPI/System/IO/outputs HTTP/1.1
Host: 172.8.6.228
Authorization: Basic YWRtaW46MTIzNDU=
```

摄像机返回的信息是一个报警输出列表，具体响应信息如下：

```
HTTP/1.1 200 OK
Server: App-webs/
Connection: close
Content-Length: 883
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<IOOutputPortList >
<IOOutputPort version="1.0" >
<id>1</id>
<PowerOnState>
<defaultState>low</defaultState>
<outputState>pulse</outputState>
<pulseDuration>30000</pulseDuration>
</PowerOnState>
</IOOutputPort>
<IOOutputPort >
<id>2</id>
<PowerOnState>
<defaultState>low</defaultState>
<outputState>pulse</outputState>
<pulseDuration>5000</pulseDuration>
</PowerOnState>
</IOOutputPort>
<IOOutputPort>
```

```
<id>3</id>
<PowerOnState>
<defaultState>low</defaultState>
<outputState>pulse</outputState>
<pulseDuration>5000</pulseDuration>
</PowerOnState>
</IOOutputPort>
</IOOutputPortList>
```

<IOOutputPortLis>是报警输出信息列表，这个列表列出 3 个报警输出口的配置信息，每个告警输出口配置信息一个<IOOutputPort>表示。<outputState>为”pulse”表示当有事件触发报警输出时，报警输出将输出一个脉冲，脉冲的宽度由<pulseDuration>指定，单位是毫秒。

也有可能设备不支持报警输出，如果是这样，<IOOutputPortList>将会是一个空表，返回的信息将会是这样的：

```
<?xml version="1.0" encoding="UTF-8"?>
<IOOutputPortList version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
</IOOutputPortList>
```

2.2.3 获取音频通道的数量

命令：GET /ISAPI/System/Audio/channels

示例如下：

```
GET /ISAPI/System/Audio/channels HTTP/1.1
Host: 172.8.6.155
Connection: Keep-Alive
Authorization: Basic YWRtaW46MTIzNDU=
```

摄像机返回的信息是音频通道的列表，具体响应信息如下:

```
HTTP/1.1 200 OK
Connection: close
Content-Length: 269
Content-Type: application/xml
```

```
<?xml version="1.0" encoding="UTF-8"?>
<AudioChannelList version="2.0"
xmlns="http://www.isapi.org/ver20/XMLSchema">
<AudioChannel version="2.0"
xmlns="http://www.isapi.org/ver20/XMLSchema">
<id>1</id>
<enabled>true</enabled>
</AudioChannel>
</AudioChannelList>
```

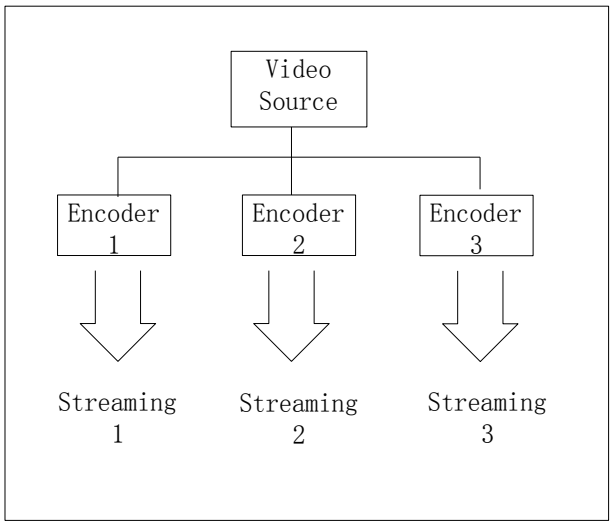
<AudioChannelList>是一个音频通道列表，对于 IPC 和快球而言当前只支持一个音频通道，所以<AudioChannelList>中只有一个<AudioChannel>。

某些类型的设备，比如 DS-2CD7153-E,是没有音频通道的，这样的设备返回的 AudioChannelList 中没有 AudioChannel：

```
<?xml version="1.0" encoding="UTF-8"?>
<AudioChannelList version="1.0"
xmlns="http://www.isapi.org/ver20/XMLSchema">
</AudioChannelList>
```

2.2.4 获取视频通道的数量

视频通道即指的视频源，有多少个视频通道就有多少个视频源，一个视频源可以进行多路码流编码，输出多路码流，它们之间的关系可以做如下理解：



命令： GET /ISAPI/System/Video/inputs/channels

示例如下：

```
GET /ISAPI/System/Video/inputs/channels HTTP/1.1
Host: 172.8.6.155
```

```
Authorization: Basic YWRtaW46MTIzNDU=
```

当前摄像机一般只支持一个视频通道，返回的视频通道列表中一般只包含一个视频通道：

```
HTTP/1.1 200 OK
Connection: close
Content-Length: 348
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<VideoInputChannelList version="2.0"
xmlns="http://www.isapi.org/ver20/XMLSchema">
<VideoInputChannel version="2.0"
xmlns="http://www.isapi.org/ver20/XMLSchema">
<id>1</id>
<inputPort>1</inputPort>
<name>TV-IP311PI</name>
<videoFormat>NTSC</videoFormat>
</VideoInputChannel>
</VideoInputChannelList>
<VideoInputChannelList>
```

中会列出当前设备上的所有通道，上面的例子是一款设备型号为 DS-2CD864FWD-E 的摄像机发回的应答，这台摄像机上只有一个视频通道，当前的制式<videoFormat>是”NTSC”。

对于 IPC 和快球，设备至少要有一个视频通道存在。

2.2.5 是否支持机械云台

摄像机通过 458 串口连接云台，如果设备上有 485 串口可以认为设备支持云台控制，获取串口列表，判断是否有 485 串口即可以判断是否支持机械云台,可以使用 index 命令，查询是否存在 485 串口资源。

命令：GET /ISAPI/System/Serial/ports/1/index

设备的返回信息如下所示：

```
<?xml version="1.0" encoding="UTF-8"?>
<ResourceList version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
<Resource version="1.0">
```

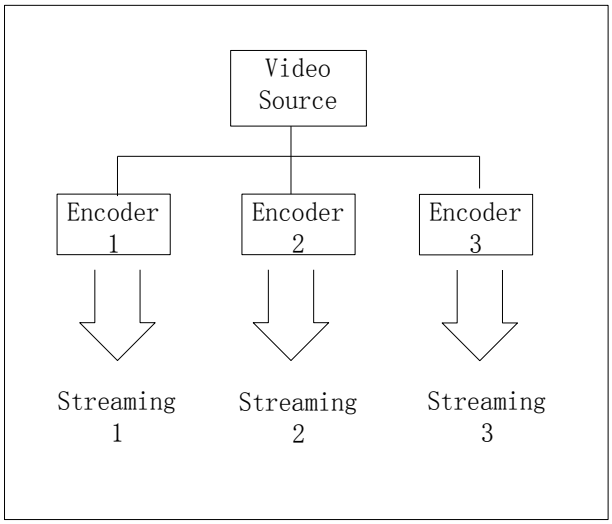
```
    … …
</ResourceList>
```

如果设备成功返回 HTTP OK，那么说明设备有 485 串口，可以支持机械云台，否则，设备将会返回错误，如下所示：

```
  <?xml version="1.0" encoding="UTF-8"?>
<ResponseStatus version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
  <requestURL>/ISAPI/Serial/ports/1/index</requestURL>
  <statusCode>4</statusCode>
  <statusString>Invalid Operation</statusString>
  </ResponseStatus>
```

3 码流配置(音视频配置)

设备可以有多个视频通道和音频通道，每个视频通道就是一个视频源，一个视频源可以有多个码流。



如何获取视频通道数量和音频通道数量已经在 2.2.4 和 2.2.3 小节中详细描述，这里不再赘述。

3.1 获取设备码流配置

对于拥有视频源的设备可能产生多个视频码流，以应用于不同需求场景，所以第一步要确认设备上当前的码流数量。

3.1.1 设备提供的码流数量

可以使用/ISAPI/Streaming/channels 命令访问设备上所有的码流，使用 GET /ISAPI/Streaming/channels 命令，设备会返回当前的所有码流信息。

示例如下：

```
GET /ISAPI/Streaming/channels HTTP/1.1
Host: 172.8.6.228
Connection: Keep-Alive
Authorization: Basic YWRtaW46MTIzNDU=
```

设备返回的响应信息如下所示：

```
HTTP/1.1 200 OK
Date: Sat, 14 Jul 2012 07:50:24 GMT
Server: App-webs/
Connection: close
Content-Length: 2802
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<StreamingChannelList version="1.0"
xmlns="http://www.isapi.org/ver20/XMLSchema">
<StreamingChannel >
<id>1</id>
<channelName>Camera 01</channelName>
<enabled>true</enabled>
<Transport>
  ... ...
</Transport>
<Video>
  ... ...
</Video>
<Audio>
  ... ..
</Audio>
</StreamingChannel>
<StreamingChannel>
```

```
<id>2</id>
<channelName>Camera 01</channelName>
<enabled>true</enabled>
<Transport>
  ... ...
</Transport>
<Video>
 ... ...
</Video>
<Audio>
 ... ...
</Audio>
</StreamingChannel>
</StreamingChannelList>
```

<StreamingChannelList>列表中列出了当前设备上支持的所有码流，每条码流用一个<StreamingChanne>表示，上面的示例中显示当前设备支持两条码流。

3.1.2 码流参数详解

在确定了设备提供的码流数量之后，可以访问某路码流，先了解一下 ISAPI 中码流中的各项参数。

使用 GET /ISAPI/Streaming/channels/1(或者/ISAPI/Streaming/channels/101)可以访问第一路码流，/ISAPI/Streaming/channels/101 这种格式是为了兼容 DVR，对于 DVR 而言有多个通道，每个通道都有若干码流，因此码流的 ID 号的排序为：通道号+通道码流号，比如第一个通道的第一路码流是对应 ID 是 101，,第一个通道的第二路码流 ID 为 102。

这里使用 GET /ISAPI/Streaming/channels/101 获取第一路码流，示例如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<StreamingChannel version="1.0"
xmlns="http://www.isapi.org/ver20/XMLSchema">
<id>1</id>
<channelName>Camera 01</channelName>
<enabled>true</enabled>
<Transport>
<maxPacketSize>1000</maxPacketSize>
<sourcePortNo>8200</sourcePortNo>
```

```
<ControlProtocolList>
<ControlProtocol>
<streamingTransport>RTSP</streamingTransport>
</ControlProtocol>
</ControlProtocolList>
<Unicast>
<enabled>true</enabled>
</Unicast>
<Multicast>
<enabled>true</enabled>
<destIPAddress>0.0.0.0</destIPAddress>
<destPortNo>8600</destPortNo>
</Multicast>
</Transport>
<Video>
<enabled>true</enabled>
<videoInputChannelID>1</videoInputChannelID>
<videoCodecType>H.264</videoCodecType>
<videoScanType>progressive</videoScanType>
<videoResolutionWidth>1600</videoResolutionWidth>
<videoResolutionHeight>1200</videoResolutionHeight>
<videoQualityControlType>CBR</videoQualityControlType>
<constantBitRate>8192</constantBitRate>
<fixedQuality>60</fixedQuality>
<maxFrameRate>3000</maxFrameRate>
<keyFrameInterval>25</keyFrameInterval>
<BPFrameInterval>0</BPFrameInterval>
<snapShotImageType>JPEG</snapShotImageType>
</Video>
<Audio>
<enabled>true</enabled>
<audioInputChannelID>11</audioInputChannelID>
<audioCompressionType>G.711ulaw</audioCompressionType>
</Audio>
</StreamingChannel>
```

码流参数是非常重要的参数，这里进行详细的说明。

<StreamingChannel>的定义在协议文档 8.9.3 小节，码流参数主要有三个部分：传输控制部分——<Transport>；视频信息部分——<Video>；音频信息部分——<Audio>。

1.传输控制部分

<Transport>在<StreamingChannel>中被定义为”req”，说明这一部分信息时必须的，无论在 GET 还是 PUT 命令都应该有这一部分信息。

<Transport>中的< ControlProtocolList >指示当前的流传输协议，这个信息可以配置。

<Transport>中的<Multicast>是关于多播的配置信息，通过改变<destIPAddress>和<destPortNo>可以配置多播地址和多播端口号。

注：在实际操作过程中，为了方便第三方集成，<Transport>信息也作为可选的信息。

2.Video 部分

<Video>部分是视频参数部分，该部分参数比较重要。下面介绍<Video>部分各参数含义：

<videoCodecType>表示的是编码格式，协议规定的格式有 H.264,MPEG4，MJPEG。

<videoScanType>是图像扫描类型，一般有逐行扫描（progressive）和隔行扫描（interlaced）。

<videoResolutionWidth>和<videoResolutionHeight>分别表示分辨率的宽和高，单位是像素。

<videoQualityControlType> 是码率控制类型，可以是定码率（CBR）和变码率（VBR）。如果是定码率，<constantBitRate>表示的是定码率的大小，单位是 kbps，设备将以恒定的码率进行编码；如果是变码率，码率的大小不固定，这时候<constantBitRate>表示的是码率的上限。

<fixedQuality>是图像质量，值为 100 时图像质量最高，值为 1 时图像质量最低。

<maxFrameRate>的值是当前码流的帧率，它的值被规定为整形，为了能够表示低于 1 帧的帧率<maxFrameRate>实际上是当前帧率乘以 100，比如如果码流的帧率是 1/4 fps，<maxFrameRate>的值就是 25；如果码流的帧率是 12fps，

<maxFrameRate>的值就是 1200。

<keyFrameInterval>表示的是 I 帧间隔，也就是每隔多少帧会有一个关键帧。

3.Audio 部分

<Audio>是码流的音频部分，对于不支持音频的设备<Audio>部分有可能不存在（如果判别设备是否支持音频通道请参考 2.2.3 小节）

如果是一款支持音频通道的设备，码流中就可以支持音频流，如果想在码流中打开音频流，可以将<enabled>设置成 true，这条码流就会有视频流和音频流。如果要关闭码流中的音频，可以将<enabled>设置成 false，这条码流中就只有视频流。

<audioCompressionType>是音频编码格式，协议定义的类型有 G.711ulaw, G.711alaw , G.726 , AAC，G722。

3.2 设置码流配置

如 3.1.2 小节描述，码流配置分为三个部分，<Transport>部分，<Video>部分和<Audio>部分。在设置码流之前，首先需要知道各个参数的可选项，比如如果要设置<Video>中的分辨率，就需要首先知道当前的设备能支持的分辨率都有哪些，获取到可选信息之后再进行设置，如果设置的信息不是设备支持的范围，设备会返回出错。

3.2.1 获取码流参数可选项

ISAPI 码流参数命令，即/ISAPI/Streaming/channels/ID 命令可以支持 capabilities 命令，在某一条资源路径后面加上 capabilities，表示访问该资源中各参数的可选项，capabilities 只支持使用 GET 方法进行获取，它表示资源的能力，是不可以被设置的。

这里使用 GET /ISAPI/Streaming/channels/101/capabilities 获取第一路码流的能力信息，示例如下：

```
GET /ISAPI/Streaming/channels/1/capabilities HTTP/1.1
Host: 172.8.6.155
Connection: Keep-Alive
Authorization: Basic YWRtaW46MTIzNDU=
```

GET /ISAPI/Streaming/channels/1/capabilities 访问的是它的主码流的能力信息，

设备返回的信息如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<StreamingChannel version="1.0"
xmlns="http://www.isapi.org/ver20/XMLSchema">
<id opt="1,2">1</id>
<channelName min="0" max="32">Camera 01</channelName>
<enabled opt="true">true</enabled>
<Transport>
<maxPacketSize opt="1000">1000</maxPacketSize>
<sourcePortNo min="0" max="65535" def="8200">8200</sourcePortNo>
<ControlProtocolList>
<ControlProtocol>
<streamingTransport opt="RTSP">RTSP</streamingTransport>
</ControlProtocol>
</ControlProtocolList>
<Unicast>
<enabled opt="true" def="true">true</enabled>
</Unicast>
<Multicast>
<enabled opt="true" def="true">true</enabled>
<destIPAddress min="8" max="16">224.1.2.3</destIPAddress>
<destPortNo min="0" max="65535" def="8600">8600</destPortNo>
</Multicast>
</Transport>
<Video>
<enabled opt="true">true</enabled>
<videoInputChannelID opt="1">1</videoInputChannelID>
<videoCodecType opt="H.264,MPEG4">H.264</videoCodecType>
<videoScanType opt="progressive">progressive</videoScanType>
<videoResolutionWidth
opt="640*480,1280*720,1280*960">1280</videoResolutionWidth>
<videoResolutionHeight
opt="640*480,1280*720,1280*960">960</videoResolutionHeight>
<videoQualityControlType opt="CBR,VBR">CBR</videoQualityControlType>
<constantBitRate min="32" max="16384">4096</constantBitRate>
<fixedQuality opt="1,20,40,60,80,100">60</fixedQuality>
```

```
  <maxFrameRate
opt="2500,2200,2000,1800,1600,1500,1200,1000,800,600,400,200,100,50,25,12,6">
600</maxFrameRate>
  <keyFrameInterval min="1" max="400">12</keyFrameInterval>
  <BPFrameInterval opt="0,1,2">0</BPFrameInterval>
  <snapShotImageType opt="JPEG" def="JPEG">JPEG</snapShotImageType>
  </Video>
  <Audio>
  <enabled opt="true,false">true</enabled>
  <audioInputChannelID opt="11">11</audioInputChannelID>
  <audioCompressionType opt="G.711ulaw"
def="G.711ulaw">G.711ulaw</audioCompressionType>
  </Audio>
    </StreamingChannel>
```

能力信息以“属性”的形式出现在各个标签中。

如果某一项参数可选的值是一个连续的区间，使用 min~max 表示，min 表示的是最小值，max 表示的是最大值，比如<constantBitRate min="32" max="16384">，它表示码率的最小值是 32kbps，最大值是 16384kbps，在此之间的任意整数值都是可以支持的。

如果某一项参数可选的值是离散的，用 opt 列出所有的可选项，比如<fixedQuality opt="1,20,40,60,80,100">这说明图像质量有 6 个等级（level），等级 1 对应的数值是 1，等级 2 是 20，等级 6 是 100。再比如<videoCodecType opt="H.264,MPEG4">，它表示主码流可以支持 H.264 编码和 MPEG4 编码两种类型。

需要特别说明视频分辨率<videoResolutionWidth>和<videoResolutionHeight>的属性中都列出了分辨率的可选项，这些可选项都是以“宽*高”的形式出现的，比如<videoResolutionHeight opt="640*480,1280*720,1280*960">，这说明主码流有三个分辨率可选项，第一个是宽 640 高 480，第二个是宽 1280 高 720，第三个是宽 1280 高 960。

3.2.2 设置码流参数

由 GET /ISAPI/Streaming/channels/ID/capabilities 获取到码流的能力之后，根据能力信息选择某些选项对码流进行配置。配置码流使用 PUT /ISAPI/Streaming/channels/ID 命令。

进行配置的时候需要注意 ISAPI 协议文档中的定义，如果某一个标签被定义

为”req”，那么这个标签必须 PUT 命令的 HTTP body 中出现，否则将视为非法。比如<Video>部分，其中的四个标签<videoResolutionWidth>，<videoResolutionHeight>，<videoQualityControlType>，<maxFrameRate>被定义为”req”，那么如果 PUT 命令中使用了<Video>，<Video>中就必须包含这四项参数，否则将视为非法。

在此给出以一个示例，使用的设备与 3.2.1 小节中的设备一样，这是一台支持音频通道的设备，这里准备把分辨率修改成 640*480，码率降低到 2048kbps，把帧率提高到 25 帧，其他参数不做修改，那么发出的 XML 内容应当是这样的：

```
<?xml version="1.0" encoding="UTF-8"?>
<StreamingChannel version="1.0"
xmlns="http://www.isapi.org/ver20/XMLSchema">
<id>1</id>
<enabled>true</enabled>
<Transport>
</Transport>
<Video>
<videoInputChannelID>1</videoInputChannelID>
<videoResolutionWidth>640</videoResolutionWidth>
<videoResolutionHeight>480</videoResolutionHeight>
<videoQualityControlType>CBR</videoQualityControlType>
<constantBitRate>2048</constantBitRate>
<maxFrameRate>2500</maxFrameRate>
</Video>
</StreamingChannel>
```

设备返回的信息如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<ResponseStatus version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
<requestURL>/streaming/channels/1</requestURL>
<statusCode>1</statusCode>
<statusString>OK</statusString>
<subStatusCode>ok</subStatusCode>
</ResponseStatus>
```

这表示请求已经执行成功，并且立即生效。

需要注意的是，改变某些参数可能需要重启生效，这时候设备返回的信息会有

相应的提示。

4 系统维护

系统维护部分主要包括设备信息配置，重启，恢复默认值，升级，系统时间操作。

4.1 设备信息配置

设备信息部分已经在 2.1 小节有所描述，设备信息中的大部分信息时只读的，比如设备的型号，设备序列号，设备描述信息，能够改变的信息目前只有设备名称和设备 Id（用于支持遥控器的设备）可以修改。

示例：设置设备名称为 NewName。

```
PUT /ISAPI/System/deviceInfo HTTP/1.1
Authorization: Basic YWRtaW46MTIzNDU=
Content-Type:text/xml
Content-Length:160

<?xml version="1.0" encoding="UTF-8"?>
<DeviceInfo version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
<deviceName>NewName</deviceName>
</DeviceInfo>
```

4.2 设备重启

设备重启命令不需要任何参数信息，设备在接收到请求之后如果执行成功会返回状态成功，然后立即重启，重启的时间依设备具体情况而定。

示例： 远程重启设备

```
PUT /ISAPI/System/reboot HTTP/1.1
Host:172.8.6.155
Authorization: Basic YWRtaW46MTIzNDU=
```

设备收到请求后先返回 OK：

```
<?xml version="1.0" encoding="UTF-8"?>
<ResponseStatus version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
<requestURL>/ISAPI/system/reboot</requestURL>
```

```
<statusCode>1</statusCode>
<statusString>OK</statusString>
</ResponseStatus>
```

然后设备立即重启。

4.3 恢复默认值

恢复默认值操作有两种方式，一种是完全恢复，一种是恢复基本参数，完全恢复会将所有的信息全部恢复到出厂默认值；而基本恢复会保留一部分客户已经完成的配置信息，比如制式、网络参数、镜头聚焦值、语言信息等，其他参数恢复成默认值。

恢复方式参数是 URI 中的，参数的名称是 mode，参数的值可以是 full 或者 basic。

示例：完全恢复默认参数

```
PUT /ISAPI/System/factoryDefault?mode=full HTTP/1.1
Host:172.8.6.155
Authorization: Basic YWRtaW46MTIzNDU=
```

示例：恢复基本参数

```
PUT /ISAPI/System/factoryDefault?mode=basic HTTP/1.1
Host:172.8.6.155
Authorization: Basic YWRtaW46MTIzNDU=
```

恢复默认参数会提示重启，比如恢复基本参数后，设备会返回：

```
HTTP/1.1 200 OK
Connection: close
Content-Length: 257
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<ResponseStatus version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
<requestURL>/system/factorydefault</requestURL>
<statusCode>7</statusCode>
<statusString>Reboot Required</statusString>
</ResponseStatus>
```

4.4 升级

升级操作用于将设备的 firmware 升级到新的版本，使用升级命令之前需要首先知道升级包的长度，升级过程中不应对设备再进行其他操作。

使用 PUT /ISAPI/System/firmwareUpgrade 命令，HTTP body 部分就是升级包，升级包内容发送完毕之后，设备会对升级包进行校验，成功之后会返回执行成功需要重启的提示，然后客户应当使用重启命令对设备进行重启（参见 4.2 小节）。

示例：升级 IPC，升级包长度 13481620 字节

```
PUT /ISAPI/System/firmwareUpgrade HTTP/1.1
HOST: 172.8.6.155
Authorization: Basic YWRtaW46MTIzNDU=
Content-Length: 13481620
Connection: keep-alive
Content-Type: application/octet-stream

.....y..
.............F\T@4JAEC.)5",EF\T@4.....m...7.....`.....
.....H...`.....2....*.....#.....
.....e.....
.....I.....|<.....Y.....9.....0..
.....p..._.....Y
..g.....(.....gK.....jB.=....5
.....+{.....u.....?z..i...d?.....B
l.....qM.....3....t..&.....I.v.....gy.....
.....+...'..V...sO.5...7T.....Q.....Linux-2.6.18_pro500-davinci_evm-.....
.....(o.....T7...p..... .....V4... ... ....!.....~0....P.
```

这里没有把升级包的所有内容全部列出来，只截取其中一部分，在 HTTP 头部结束之后，就是升级包的内容，升级包的内容是二进制格式，所以在 HTTP 头部中 Content-Type 被指定为 application/octet-stream(也可以指定为 application/binary)。

4.5 系统时间操作

ISAPI 提供了两种校时方式：NTP 校时和手动校时。

4.5.1 NTP 校时

如果设备设置成 NTP 校时，设备会向校时服务器发送请求，校时服务器向设备返回 UTC 时间，设备根据 UTC 时间进行校时。

设置 NTP 校时需要首先设置 NTP 服务器，然后将系统校时模式设置为 NTP 校时模式。

1. 设置 NTP 服务器

如果设备支持 NTP 校时，使用 GET /ISAPI/System/time/ntpServers 命令，设备的应答中至少会包含一个 NTP 服务器的配置，如下所示：

```
GET /ISAPI/System/time/ntpServers HTTP/1.1
Host: 172.8.6.155
Authorization: Basic YWRtaW46MTIzNDU=
```

设备返回的信息如下所示：

```
<?xml version="1.0" encoding="UTF-8"?>
<NTPServerList version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
<NTPServer version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
<id>1</id>
<addressingFormatType>hostname</addressingFormatType>
<hostName></hostName>
<portNo>123</portNo>
</NTPServer>
</NTPServerList>
```

返回的信息是一个 NTP 服务器配置列表，当前的设备支持配置 1 个 NTP 服务器，上例中没有对服务器地址进行配置，所以校时服务器的名称<hostName>是空的。

NTP 服务器的地址可以由两种方式表示：域名或者 IP 地址。

如果使用域名，<addressingFormatType>的值应当指定为”hostname”，选定了域名方式，那么<hostName>这个标签就是必须的，它指定了域名。

如果使用 IP 地址，<addressingFormatType>的值应当指定为”ipaddress”，选定了 IP 地址方式，那么<ipAddress>这个标签就是必须的，它指定了 ip 地址。

下面给出一个设置 NTP 服务器的例子，设置 NTP 服务器方式为指定 IP，示例如下：

```
PUT /ISAPI/System/time/ntpServers/1 HTTP/1.1
Authorization: Basic YWRtaW46MTIzNDU=
Content-Type: text/xml
Host: 172.8.6.155
Content-Length: 266

<?xml version="1.0" encoding="utf-8"?>
<NTPServer>
<id>1</id>
<addressingFormatType>ipaddress</addressingFormatType>
<ipAddress>10.6.8.238</ipAddress>
<portNo>123</portNo>
<Extensions>
<synchronizeInterval>60</synchronizeInterval>
</Extensions>
</NTPServer>
```

NTP 服务器的地址被指定为 10.6.8.238，服务器端口号为 123。默认情况下，校时的间隔为 24 小时一次，部分新的设备支持设置校时间隔，校时间隔放在了扩展信息部分，用<synchronizeInterval>表示，单位是分钟，上例中，设置的校时间隔是 60 分钟。

2. 设置系统校时模式

设置完 NTP 校时服务器之后，可以指定系统进行 NTP 校时。使用 PUT /ISAPI/System/time 命令修改当前的校时模式。

示例：修改当前校时模式为 NTP 校时

```
PUT /ISAPI/System/time HTTP/1.1
Authorization: Basic YWRtaW46MTIzNDU=
Content-Type: text/xml; charset=utf-8
Host: 172.8.6.155
Content-Length: 115

<?xml version="1.0" encoding="utf-8"?>
<Time>
<timeMode>NTP</timeMode>
```

```
<timeZone>CST-8:00:00</timeZone>
</Time>
```

上述命令设置校时模式为 NTP 校时，设置当前时区为东 8 区（CST-8:00:00），假如设置生效后系统立即校时（实际上取决于校时间隔），校时服务器返回的时间是 12 点整，那么设备将根据当前的时区加上 8 个小时，校时后的本地时间为 2012 年 7 月 15 日 20 点整。

4.5.2 手动校时

手动校时需要首先将时间模式设置为手动方式，并指定本地时间。

示例：设置手动校时

```
PUT /ISAPI/System/time HTTP/1.1
Authorization: Basic YWRtaW46MTIzNDU=
Content-Type: text/xml; charset=utf-8
Host: 172.8.6.155
Content-Length: 162

<?xml version="1.0" encoding="utf-8"?>
<Time>
<timeMode>manual</timeMode>
<localTime>2012-07-15T19:40:04</localTime>
<timeZone>CST-8:00:00</timeZone>
</Time>
```

设置成手动校时之后，系统时间将以<localTime>指定的时间为准并立即生效。

5 IO 控制

IO 控制部分指的是对报警输入和报警输出进行配置，并非所有的设备都有报警输入和报警输出，因此在进行 IO 配置之前需要获取设备的硬件能力信息（请参考 2.2.1 小节和 2.2.2 小节）

5.1 报警输入控制

可以设置报警输入的有效信号形式，即是高电平触发报警输入还是低电平触发报警输入。比如设置高电平触发报警输入，那么非报警状态下，报警端口处于

断开状态，一旦闭合就会产生报警。

示例：设置报警输入端口 1，设置为高电平触发报警输入

```
PUT /ISAPI/System/IO/inputs/1 HTTP/1.1
Authorization: Basic YWRtaW46MTIzNDU=
Host:172.8.6.155
Content-Type:text/xml
Content-Length:171

<?xml version="1.0" encoding="UTF-8"?>
<IOInputPort version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
<id>1</id>
<triggering>high</triggering>
</IOInputPort>
```

5.2 报警输出控制

当某个条件产生，设备判断需要触发报警输出时，报警输出端口会产生一个脉冲，脉冲的宽度可以被指定。

示例：设置报警输出端口 1，设置报警输出脉冲宽度为 0.5 秒

```
PUT /ISAPI/System/IO/outputs/1 HTTP/1.1
Authorization: Basic YWRtaW46MTIzNDU=
Content-Type:text/xml
Content-Length:280

<?xml version="1.0" encoding="UTF-8"?>
<IOOutputPort version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
<id>1</id>
<PowerOnState>
<defaultState>low</defaultState>
<outputState>pulse</outputState>
<pulseDuration>5000</pulseDuration>
</PowerOnState>
</IOOutputPort>
```

报警输出的脉冲宽度被指定为 5000 毫秒，即 0.5 秒。

5.3 手动触发报警输出

报警输出一般作为一种联动方式，当事件发生时由设备内部逻辑判断并触发

报警输出。除此之外，报警输出也可以手动进行触发，需要首先设置报警输出模式为手动触发，然后调用触发命令，强制报警输出产生，或者关闭。

5.3.1 设置报警输出为手动触发模式

使用 PUT /ISAPI/System/IO/outputs/ID 模式设置报警输出工作在手动触发模式，5.2 小节中列举的例子是设置为脉冲模式，在脉冲模式下，事件发生后如果联动方式设置为报警输出，那么报警输出口将产生一个脉冲；如果要设置报警输出为手动触发模式，只有调用手动触发报警输出命令：PUT /ISAPI/System/IO/outputs/ID/trigger，报警输出才会输出高电平。

可以通过 PUT /ISAPI/System/IO/outputs/ID 将 outputState 设置为 high，将报警输出设置在手动触发的模式下。

示例：设置报警输出为手动触发模式

```
PUT /ISAPI/System/IO/outputs/1
Authorization: Basic YWRtaW46MTIzNDU=
Content-Type:text/xml
Content-Length:245

<?xml version="1.0" encoding="UTF-8"?>
<IOOutputPort version="1.0"
xmlns=" http://www.isapi.org/ver20/XMLSchema">
<id>1</id>
<PowerOnState>
<defaultState>low</defaultState>
<outputState>high</outputState>
</PowerOnState>
</IOOutputPort>
```

5.3.2 手动触发报警输出

使用 PUT /ISAPI/System/IO/outputs/1/trigger 命令，可以手动触发报警输出产生一个高电平。

示例：手动触发报警输出

```
PUT /ISAPI/System/IO/outputs/1/trigger
Authorization: Basic YWRtaW46MTIzNDU=
Content-Type:text/xml
```

```
Content-Length:246

<IOPortData xmlns="http://www.isapi.org/ver20/XMLSchema">
<outputState>high</outputState>
</IOPortData>
```

示例说明：该命令强制在报警输出产生一个高电平，如果要结束手动触发，可以将<outputState>的值置为 low。

5.4 报警输入事件

ISAPI 协议支持事件功能，可以指定告警输入的联动方式，当告警输入发生时将会完成指定的联动方式，比如可以配置告警输入触发 Email 发送，当告警发生时将会发送邮件到指定的地址。

关于此部分功能请参考《ISAPI 事件功能开发文档》。

6 网络配置

设备有可能支持多个网络界面，比如对于 IPC 而言，可以同时存在有线网络和无线网络，在这种情况下一般第一个网络界面指的是有线网络，第二个网络界面指的是无线网络。对于非 WIFI 的设备一般情况下只有一个网络界面，即 /Network/interfaces/1。

6.1 网络配置

ISAPI 协议提供了配置网络界面的接口，配置的参数包括 IP 地址，掩码，网关，DNS 服务器等网络界面信息。

示例：配置有线网络接口网络信息。

```
PUT /ISAPI/System/Network/interfaces/1/ipAddress HTTP/1.1
Authorization: Basic YWRtaW46MTIzNDU=
Host:172.8.6.155
Content-Type:text/xml
Content-Length:886

<IPAddress version="2.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
<ipVersion>dual</ipVersion>
<addressingType>dynamic</addressingType>
```

```
<ipAddress>172.8.6.155</ipAddress>
<subnetMask>255.255.255.0</subnetMask>
<ipv6Address>3000:1:2:4:8ee7:48ff:fedb:fae</ipv6Address>
<bitMask>64</bitMask>
<DefaultGateway>
<ipAddress>172.8.6.1</ipAddress>
<ipv6Address>::</ipv6Address>
</DefaultGateway>
<PrimaryDNS>
<ipAddress>10.1.7.88</ipAddress>
</PrimaryDNS>
<SecondaryDNS>
<ipAddress>10.1.7.77</ipAddress>
</SecondaryDNS>
<Ipv6Mode>
<ipV6AddressingType>ra</ipV6AddressingType>
<ipv6AddressList>
<v6Address>
<id>1</id>
<type>ra</type>
<address>3000:1:2:4:8ee7:48ff:fedb:fae</address>
<bitMask>64</bitMask>
</v6Address>
<v6Address>
<id>2</id>
<type>manual</type>
<address>::</address>
<bitMask>0</bitMask>
</v6Address>
</ipv6AddressList>
</Ipv6Mode>
</IPAddress>
```

上例中使用的网络界面 ID 是 1，这是有线网络的网络界面 ID，如果是无线网络（设备支持 WIFI 的情况下），网络界面 ID 是 2.

将<addressingType>指定为静态（static），这样配置的 IP 地址，网关，DNS 服务器等信息才生效，否则如果指定为”danymic”则会启用 DHCP 功能，通过

DHCP 服务器获取网络参数。

<ipVersion>指示的是 IP 地址类型，上例中将 IP 地址类型指定为 Ipv4 地址类型。

<subnetMask> 指示的是网络掩码，示例中设置了 24 位掩码。

<DefaultGateway>指示的是默认网关，示例中设置的默认网关为”172.8.6.1”。

<PrimaryDNS>指示的是 DNS 服务器地址，示例中设置主 DNS 服务器地址为 10.1.7.88，辅 DNS 服务器地址为 10.1.7.77。

6.2 开启 DHCP

开启 DHCP 功能之后，设备的网路配置将从 DHCP 服务器获取，包括网络地址，掩码，网关信息都没有必要再配置，但是可以配置 DNS 服务器的地址。使用 ISAPI 接口开启 DHCP 功能实际上就是将地址类型设置为动态地址类型。

示例：开启 DHCP 功能

```
PUT /ISAPI/System/Network/interfaces/1/ipAddress HTTP/1.1
Authorization: Basic YWRtaW46MTIzNDU=
Host:172.8.6.155
Content-Type:text/xml
Content-Length:193

<?xml version="1.0" encoding="UTF-8"?>
<IPAddress version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
<ipVersion>v4</ipVersion>
<addressingType>dynamic</addressingType>
</IPAddress>
```

如果原来的地址类型是手动配置 IP 信息，即”static”类型，现在改为”dynamic”设备会提示重启生效：

```
HTTP/1.1 200 OK
Connection: close
Content-Length: 257
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<ResponseStatus version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
<requestURL>/system/factorydefault</requestURL>
<statusCode>7</statusCode>
<statusString>Reboot Required</statusString>
```

```
<subStatusCode>rebootRequired</subStatusCode>
</ResponseStatus>
```

6.3 DDNS 功能

DDNS 功能一般用于没有固定 IP 地址的设备，设备每次重新启动都可能会有不同的 IP 地址，比如开启 PPPoE 功能，设备会拨号上网，每次分配给设备的地址都有可能不同，这种情况下可以给设备申请一个域名，设备重启网络配置发生改变会将自己的 IP 地址和认证信息发送到 DDNS 服务器，服务器将设备的 IP 址和域名关联起来并做记录，在以后交互过程中就可以使用域名直接访问设备。

开启 DDNS 功能需要指定 DDNS 服务提供商，服务器地址和端口号，设备的域名，以及认证信息。

示例：设置 DDNS 服务

```
PUT /ISAPI/System/Network/interfaces/1/ddns HTTP/1.1
Authorization: Basic YWRtaW46MTIzNDU=
Host:172.8.6.155
Content-Type:text/xml
Content-Length:193

<?xml version="1.0" encoding="UTF-8"?>
<DDNS version="1.0" xmlns="http://www.isapi.org/ver20/XMLSchema">
<enabled>true</enabled>
<provider>DynDNS</provider>
<serverIPAddress>10.16.2.10</serverIPAddress>
<portNo>8200</portNo>
<domainName>www.hik-online.com</domainName>
<userName>ipcamera</userName>
<password>ipc1601480</password>
</DDNS>
```

示例中开启了 DDNS 服务，<enabled>被设置为”true”。

设置服务提供商为 DynDNS，设置服务器地址为 10.16.2.10，服务器端口号为 8200.

设置设备域名为” www.hik-online.com”，此域名就是动态地址要绑定的域名。

用户名和密码在 PUT 命令中是应当出现的，其中密码标注成只写“wo”，意味着密码只有在 PUT 命令中出现，在 GET 命令中是不出现的，这样做是为了保护认证信息。

7 语音对讲

对于支持语音通道的设备，可以使用 ISAPI 协议接口进行语音对讲（请参考 2.2.3 小节中的介绍）。

进行语音对讲之前，先要打开语音对讲通道，这时候设备会进行一系列初始化操作。语音对讲结束应当关闭语音对讲通道。语音对讲是双向的，既可以从设备接收语音数据，也可以向设备发送数据。

语音对讲的一般操作步骤为：

（1） 打开语音对讲通道

使用 PUT /ISAPI/System/TwoWayAudio/channels/1/open 打开设备语音对讲，此时设备会进行设备资源初始化。

（2） 接收/发送语音数据

（2.1）使用 GET /ISAPI/System/TwoWayAudio/channels/1/audioData，从设备接收语音数据。

设备接收到该命令后，将会创建一个任务，该任务将编码后的语音数据发送到客户端。该任务只允许创建一次，这意味着 GET /ISAPI/System/TwoWayAudio/channels/ID/audioData 命令只能调用一次，设备会先返回 HTTP OK 消息，HTTP Header 结束之后的所有数据都是语音数据，不再有其他数据。当连接中断，或者客户端调用了 PUT /ISAPI/System/TwoWayAudio/channels/ID/close 命令，该任务结束。

（2.2）使用 PUT /ISAPI/System/TwoWayAudio/channels/1/audioData,向设备发送语音数据。

设备接收到该命令后，会创建接收语音数据的任务，等待客户端发送的语音数据，结束语音数据的任务只能创建一次，这意味着客户端只能调用一次 PUT /ISAPI/System/TwoWayAudio/channels/1/audioData 命令，连续两次调用会返回失败。

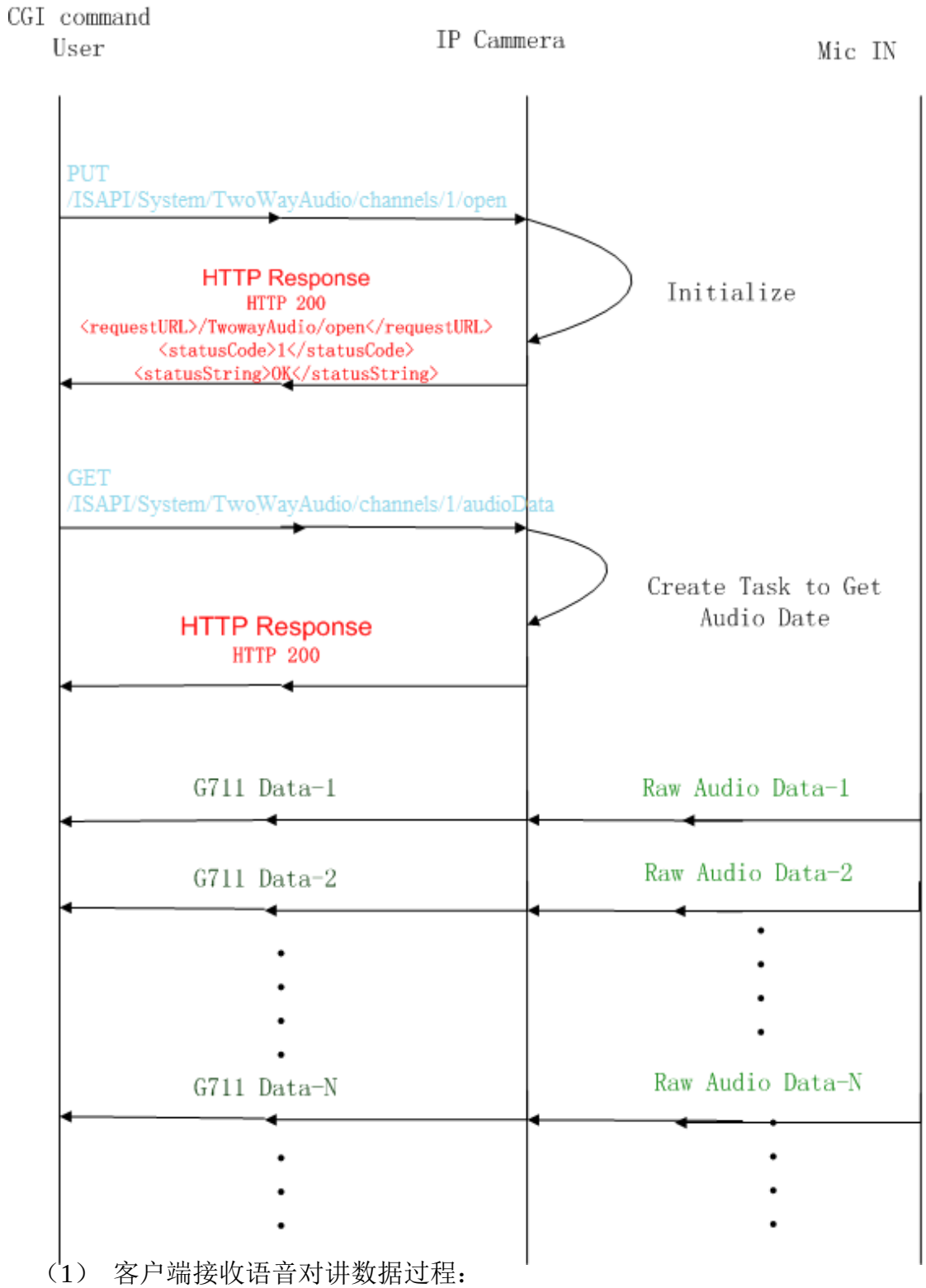
当连接中断，或者客户端调用了 PUT /ISAPI/System/TwoWayAudio/channels/ID/close 命令，该任务结束。

调用 PUT /ISAPI/System/TwoWayAudio/channels/ID/audioData 之后，如果一切正常，设备会返回 HTTP OK 消息告知客户端任务创建成功，可以向设备发送语音数据，此时客户端直接通过 TCP 连接发送语音数据即可，不必再使用 PUT 命令。

（3） 使用 PUT /ISAPI/System/TwoWayAudio/channels/ID/close 关闭语音对讲

通道。成功关闭语音对讲通道之后，设备会向客户端返回 HTTP OK 消息。使用该命令后，语音接收任务和语音发送任务会同时退出。

语音对讲的过程可以用下面的图示表示：



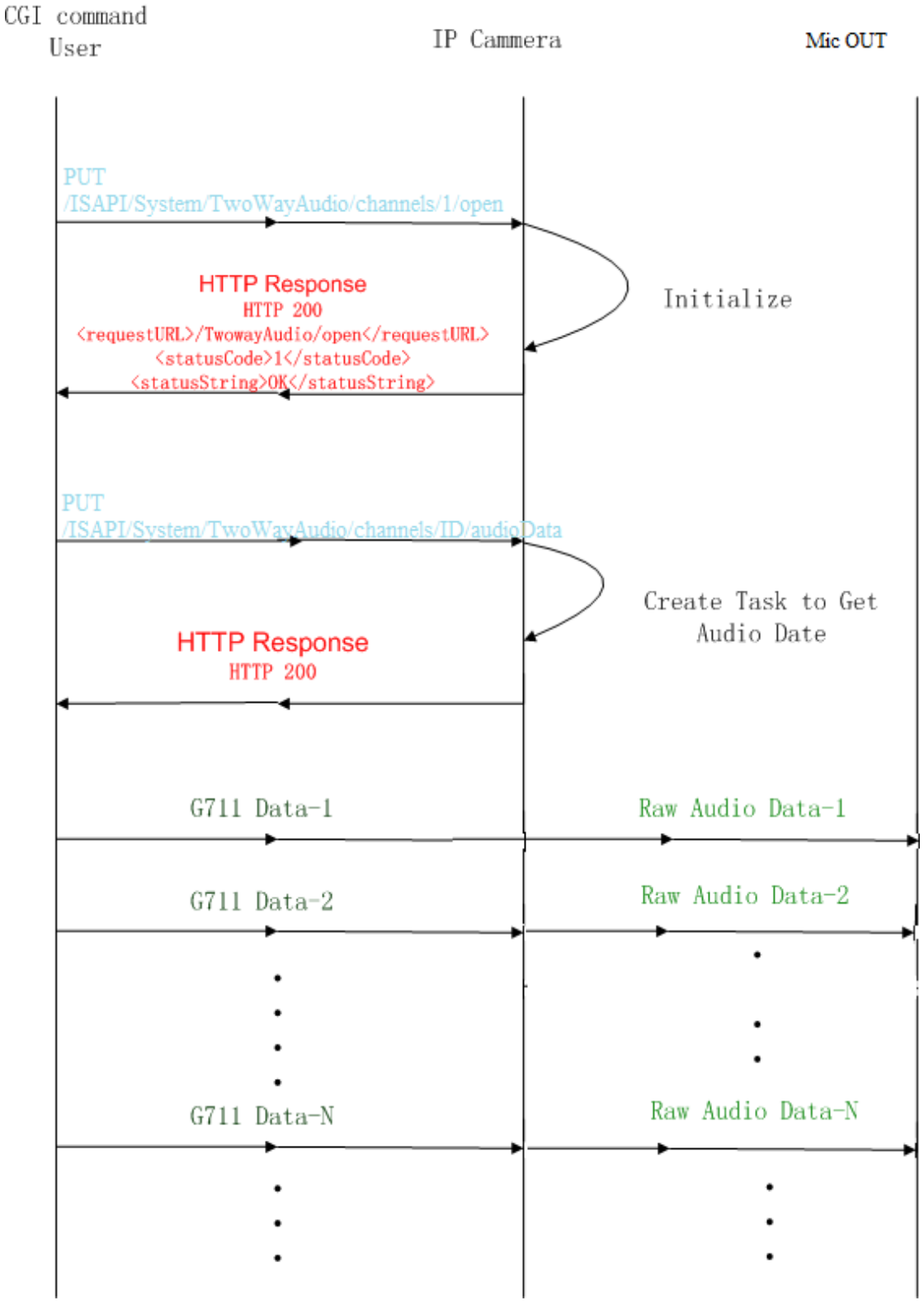
（1） 客户端接收语音对讲数据过程：

上图所示的语音对讲过程中，客户端首先使用 open 命令打开语音对讲通道，然后使用 GET 命令，设备收到 GET 命令之后创建了语音数据发送任务，任务

创建成功设备会首先返回 HTTP OK 消息告知客户端，随后任务开始执行，设备把从 Mic In 采样到的数据进行 G711 编码（编码格式取决于当前配置），然后不断发送到客户端。HTTP OK 消息只会在创建任务成功时发送一次，随后设备发送到客户端的都是语音数据，不再有其他数据。

客户端只需要调用一次 GET /ISAPI/System/TwoWayAudio/channels/ID/audioData 命令，设备会不断将编码后的语音数据发送出去，直到连接中止或 close 命令到来，如果连续两次调用 GET 语音命令获取数据，设备会返回失败。

（2）客户端发送语音数据过程：



上图所示的语音对讲过程中，客户端首先使用 open 命令打开语音对讲通道，然后使用 PUT 命令，设备收到 PUT 命令之后创建了语音数据接收任务，任务创建成功设备会首先返回 HTTP OK 消息告知客户端，随后任务开始执行，任务等待接收从客户端发送的语音数据，接收到语音数据后进行解码，然后送往 Mic Out 进行播放。

客户端在发送完 PUT /ISAPI/System/TwoWayAudio/channels/ID/audioData 指令之后，后续通过 TCP 连接直接发送编码后的语音数据，每次发送不必再使用 PUT 指令，直接发送编码语音数据即可（如图所示），PUT /ISAPI/System/TwoWayAudio/channels/ID/audioData 仅用来创建任务，该任务只可创建一次，重复创建会导致失败。

8 附录：

8.1 HTTP Header 中的 Content-Type：

application/xml : XML 格式文本内容

text/plain: 普通文本格式，比如本地时间的显示内容

application/binary：二进制文件格式，比如升级包内容

image/jpeg: JPEG 图片，比如抓图命令返回的内容

multipart/mixed: 该格式标明 http body 较长，采用分段格式，各段内容的格式有可能不一致。该格式声明一般如下：multipart/mixed; boundary= boundary1，上例中 boundary 声明了各段内容分割字符是 boundary1。比如获取事件流 GET /Event/notification/alertStream 的返回结果就采用了多段格式。

audio/basic:语音数据，比如获取语音对讲数据

application/soap+xml：SOAP 格式

application/x-x509-ca-cert: 根证书，采用 X509 算法

application/x-x509-client-cert：客户端证书

application/x-x509-client-key：客户端密钥

8.2 ResponseStatus 页面

ResponseStatus 页面一般用来指示 PUT 操作或者 POST 操作的结果，ISAPI 协议共定义了 7 种状态码来表示操作的结果，该页面的具体定义信息如下：

<?xml version="1.0" encoding="utf-8" ?>

<ResponseStatus version="1.0" xmlns="
http://www.isapi.org/ver20/XMLSchema">

<requestURL>/ISAPI/Streaming/Channels</requestURL>

<statusCode>1</statusCode>

<!-- O=1-OK, 2-Device Busy, 3-Device Error, 4-Invalid Operation, 5-Invalid XML Format, 6-Invalid XML Content; 7-Reboot Required-->

<statusString>OK</statusString>

<subStatusCode>ok</subStatusCode>

</ResponseStatus>