介绍

ONVIF 的目标是为了实现完全标准化的、可互操作性的网络视频服务，即使是由不同的网络视频供应商组成的产品。规范描述了网络视频模型，接口，数据类型和数据交换模式。规范使用了那些已经存在的的相关标准，并同时根据视频网络服务添加制定了一些必要的新规范。

这是 ONVIF 的核心规范，另外，ONVIF 已经发布了以下规范：

ONVIF 架构 [ONVIF 架构]
ONVIF 分析服务 WSDL [ONVIF 分析服务 WSDL]
ONVIF 解析设备服务 [ONVIF 解析设备服务 WSDL]
ONVIF 设备服务 WSDL [ONVIF 设备 WSDL]
ONVIF 设备 IO 服务 WSDL [ONVIF 设备 IO WSDL]
ONVIF 显示服务 WSDL [ONVIF 显示 WSDL]
ONVIF 事件服务 WSDL [ONVIF 事件 WSDL]
ONVIF 图像服务 WSDL [ONVIF 图像 WSDL]
ONVIF 媒体服务 WSDL [ONVIF 媒体 WSDL]
ONVIF PTZ 服务 WSDL [ONVIF PTZ WSDL]
ONVIF 记录服务 WSDL [ONVIF 记录 WSDL]
ONVIF 远程发现 WSDL [ONVIF 远程发现 WSDL]
ONVIF 重放服务 WSDL [ONVIF 重放 WSDL]
ONVIF 检索服务 WSDL [ONVIF 检索 WSDL]
ONVIF 主题 XML 命名空间[ONVIF 主题命名空间]

文档是按照 ONVIF 的规范框架所编写的，ONVIF 文档被分为以下几个部分：
规范的综述：把规范的各个部分给出一个概要，以及它们之间的关联性
网络服务框架：对网络服务和基于 ONVIF 的网络服务规范给出一个简要的介绍
IP 配置：规定 ONVIF 网络视频服务的 IP 配置要求
设备发现：描述设备怎样被发现，在当地或远程的网络中
设备管理：定义网络视频发射器管理命令
设备 IO：定义处理物理层的输入输出命令
图形和媒体：定义与图像和媒体设置相关的配置命令
实时流：为音视频流和元数据流提供需要
事件处理:定义怎样同意和接收数据从网络视频事件（通知）
PTZ 控制:提供命令控制云台全方位（上下、左右）移动及镜头变倍、变焦
视频分析：定义了 ONVIF 分析模式，分析对象描述和解析规则配置
视频解析设备：定义命令处理视频解析设备
记录控制：定义记录配置机制
检索和重播控制：提供用于记录的包括元数据的检索命令
安全章节：定义在 ONVIF 中数据的传输安全级别

1 范围

本标准定义的是网络视频客户端和视频传输设备的通信规范。这个新的规范使不同的厂商提供的产品均可以通过一种统一的接口进行通信成为了可能。这些接口包括功能如：设备管理、实时的音视频流、事件处理，PTZ 控制（云台全方位（上下、左右）移动及镜头变倍、变焦控制），视频分析及控制，搜索与回放。

ONVIF 规范中设备管理和控制部分所定义的接口均以 Web Services 的形式提供。为了引进网络视频服务，ONVIF 规范包含了所有的 XML 及 WSDL（Wed Service Description Language）的定义。

为了达到完全的即插即用的操作性能,该标准定义了设备发现的规范。ONVIF 中设备发现机制可看作是 WS-Discovery 的延伸。因为网络视频发现机制的需要，关于 WS-Discovery 的延伸也会在后面被讲到。

ONVIF 规范不仅具有设备的发现，配置和控制功能，而且在 IP 网络方面，ONVIF 也为媒体和元数据流定义了严格的格式，此外，还在 ONVIF 中对规范做了一定的扩展，为了就是让制造商给客户提供一个完整的网络视频传输解决方案。

2 引用标准

ISO/IEC 14496-3:2005, Information technology -- Coding of audio-visual objects -- Part 3: Audio
ISO/IEC 14496-2:2004, Information technology -- Coding of audio-visual objects -- Part 2: Visual
ISO/IEC 14496-10:2008, Information technology -- Coding of audio-visual objects -- Part 10: Advanced Video Coding
ITU-T G.711, Pulse code modulation (PCM) of voice frequencies
<http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-G.711-198811-I!!PDF-E&type=items>
ITU-T G.726, 40, 32, 24, 16 kbit/s Adaptive Differential Pulse Code Modulation (ADPCM)
<http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-G.726-199012-I!!PDF-E&type=items>
RSA Laboratories, PKCS #10 v1.7: Certification Request Syntax Standard, RSA Laboratories
<ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-10/pkcs-10v1_7.pdf>
FIPS 180-2, SECURE HASH STANDARD
<http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf>
IETF RFC 2131, Dynamic Host Configuration Protocol
<http://www.ietf.org/rfc/rfc2131.txt>
IETF RFC 2136, Dynamic Updates in the Domain Name System (DNS UPDATE)
<http://www.ietf.org/rfc/rfc2136.txt>
IETF RFC 2246, The TLS Protocol Version 1.0
<http://www.ietf.org/rfc/rfc2246.txt>
IETF RFC 2326, Real Time Streaming Protocol (RTSP)
<http://www.ietf.org/rfc/rfc2326.txt>
IETF RFC 2435, RFC2435 - RTP Payload Format for JPEG-compressed Video
<http://www.ietf.org/rfc/rfc2435.txt>
IETF RFC 2616, Hypertext Transfer Protocol -- HTTP/1.1
<http://www.ietf.org/rfc/rfc2616.txt>
IETF RFC 2617, HTTP Authentication: Basic and Digest Access Authentication
<http://www.ietf.org/rfc/rfc2617.txt>
IETF RFC 2782, A DNS RR for specifying the location of services (DNS SRV)
<http://www.ietf.org/rfc/rfc2782.txt>
IETF RFC 2818, HTTP over TLS
<http://www.ietf.org/rfc/rfc2818.txt>
IETF RFC 3268, Advanced Encryption Standard (AES) Cipher suites for Transport Layer Security (TLS)
<http://www.ietf.org/rfc/rfc3268.txt>
IETF RFC 3315, Dynamic Host Configuration Protocol for IPv6 (DHCPv6)
<http://www.ietf.org/rfc/rfc3315.txt>
IETF RFC 3548, The Base16, Base32, and Base64 Data Encodings
<http://www.ietf.org/rfc/rfc3548.txt>
IETF RFC 3550, RTP: A Transport Protocol for Real-Time Applications
<http://www.ietf.org/rfc/rfc3550.txt>

IETF RFC 3551, RTP Profile for Audio and Video Conferences with Minimal Control
<http://www.ietf.org/rfc/rfc3551.txt>
IETF RFC 3927, Dynamic Configuration of IPv4 Link-Local Addresses
<http://www.ietf.org/rfc/rfc3927.txt>
IETF RFC 3984, RTP Payload Format for H.264 Video
http://www.ietf.org/rfc/rfc3984
Universally Unique IDentifier (UUID) URN Namespace
<http://www.ietf.org/rfc/rfc4122.txt>
IETF RFC 4346, The Transport Layer Security (TLS) Protocol Version 1.1
<http://www.ietf.org/rfc/rfc4346.txt>
IETF RFC 4566, SDP: Session Description Protocol
<http://www.ietf.org/rfc/rfc4566.txt>
IETF RFC 4571, Framing Real-time Transport Protocol (RTP) and RTP Control Protocol (RTCP) Packets over
Connection-Oriented Transport
<http://www.ietf.org/rfc/rfc4571.txt>
IETF RFC 4585, Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback
(RTP/AVPF)
<http://www.ietf.org/rfc/rfc4585.txt>
IETF 4702, The Dynamic Host Configuration Protocol (DHCP) Client Fully Qualified Domain Name (FQDN) Option
<http://www.ietf.org/rfc/rfc4702.txt>
IETF 4861, Neighbor Discovery for IP version 6 (IPv6)
<http://www.ietf.org/rfc/rfc4861.txt>
IETF 4862, IPv6 Stateless Address Auto configuration
<http://www.ietf.org/rfc/rfc4862.txt>
IETF 5104, Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)
<http://www.ietf.org/rfc/rfc5104.txt>
IETF 5246, The Transport Layer Security (TLS) Protocol Version 1.2
<http://www.ietf.org/rfc/rfc5246.txt>
W3C SOAP Message Transmission Optimization Mechanism,
<http://www.w3.org/TR/soap12-mtom/>
W3C SOAP 1.2, Part 1, Messaging Framework
<http://www.w3.org/TR/soap12-part1/>
W3C SOAP Version 1.2 Part 2: Adjuncts (Second Edition)
<http://www.w3.org/TR/2007/REC-soap12-part2-20070427/>
W3C Web Services Addressing 1.0 – Core
<http://www.w3.org/TR/ws-addr-core/>
OASIS Web Services Base Notification 1.3
<http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf>
XMLSOAP, Web Services Dynamic Discovery (WS-Discovery)", J. Beatty et al., April 2005.
<http://specs.xmlsoap.org/ws/2005/04/discovery/ws-discovery.pdf>
OASIS Web Services Security: SOAP Message Security 1.1 (WS-Security 2004)

<http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessage Security.pdf>
OASIS Web Services Topics 1.3
<http://docs.oasis-open.org/wsn/wsn-ws_topics-1.3-spec-os.pdf>
OASIS Web Services Security UsernameToken Profile 1.0
<http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0.pdf>
W3C Web Services Description Language (WSDL) 1.1
http://www.w3.org/TR/wsdl
<http://www.w3.org/TR/xmlschema-1/>
W3C XML Schema Part 2: Datatypes Second Edition
<http://www.w3.org/TR/xmlschema-2/>
W3C XML-binary Optimized Packaging
<http://www.w3.org/TR/2005/REC-xop10-20050125/>
IEEE 802.11, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications
<http://standards.ieee.org/getieee802/download/802.11-2007.pdf>
IEEE 802.1X, Port-Based Network Access Control
<http://standards.ieee.org/getieee802/download/802.1X-2004.pdf>

3 术语与定义

3.1 定义

Ad-hoc network 自组织网络，经常作为一个本地独立的基本服务设置术语来使用，在[IEEE 802.11-2007].中定义
Basic Service Set 基础服务集,一组成功加入到一个公共网络中的 IEEE802.11 工作站，见[IEEE 802.11-2007]
Capability 功能命令，允许一个客户端通过设备请求服务的功能
Configuration Entity 配置实体，一个网络视频设备抽象组件，用于在网络上产生媒体流，也就是音视频流
Control Plane 控制平台，由媒体的控制功能组成如：设备控制，媒体配置和 PTZ 命令
Digital PTZ 数字 PTZ，通过调整图像的位置和比例来减小或扩大一张图像
Imaging Service 成像功能 曝光时间，高增益和白平衡参数等等功能
Infrastructure network 网络构架，一个 IEEE 802.11 网络包括一个接入口，如【IEEE 802.11】定义
Input/Output (I/O).输入/输出，一般的端口和音视频输入输出口
Layout 布局，定义在监视器上显示区域的安排
Media Entity 媒体实体，媒体配置实体例如视频源，编码器，音频源，PTZ 和解析器
Media Plane 媒体平台，由媒体流组成如音视频和元数据
Media Profile 媒体属性，管理一个音视频源或一个音频输出到一个视频或一个音频解码器，还管理音频解码器、PTZ、分析器配置
Metadata 元数据，除了音视频外的所有流数据，包括视频分析结果，PTZ 位置数据和其他元数据（如系统应用的文本数据）
Network Video Transmitter (NVT) 网络视频发射器，网络视频服务（例如一个 IP 网络摄像机

或一个解码驱动器）通过一个 IP 网络送媒体数据给客户端
Network Video Display(NVD)　网络视频显示器，网络视频接收器（例如一个网络监视器）通过 IP 网络从 NVT 接收媒体数据
Network Video Storage (NVS) 网络视频存储器，一个存储从流设备接收到的媒体数据和元数据，如一个 NVT　，通过 IP 网络传送到一个永久存储媒介。网络视频服务器也能使能客户端查看存储器中的数据
Network Video Analytics (NVA) 网络视频分析器，用于分析从流设备收到的数据，如一个 NVT 或一个存储设备例如一个 NVS
Optical Zoom 变焦　改变 NVT 的焦距
Pane 窗格在一块屏面上定义一定区域
PKCS 公钥加密标准 指的是一些 RSA 安全机构设计和发布的公钥标准
Pre Shared Key 设备静态码　设备的静态密钥
PTZ Node　PTZ 节点 低级的 PIZ 实体管理 PTZ 设备和它的功能
PullPoint 拖曳消息资源 通过拖曳消息，通知不会被防火墙阻塞
Recording 记录　表示当前的存储媒介和在 NVS 上从单一数据源接收的元数据，一个记录可以包含一个或多个轨道，一个记录能有多个同类型的轨道，如在同时记录两个具有不同设置的视频轨道
Recording Event API 记录事件　一个事件与一个记录相关联，通过一个应用接口消息表现出来
Recording Job 记录工作　通过特定地配置，让传送数据从一个数据源到指定的一条记录数据中
Remote Discovery Proxy (Remote DP) 远程设备搜索服务器 此服务器允许一台 NVT 在远程设备搜索服务器上注册，并在客户端上通过 Remote DP 找到注册的 NVTs，即使 NVC 和 NVT 在不同的网络管理域中。
Scene Description 场景描述　通过视频分析器把场景的位置和行为转换为元数据输出
Service Set ID 服务 ID 一个 IEEE802.11 无线网络服务身份号
Track 轨道 一段独有的由音视频或元数据组成的数据信道，这个定义与[RFC 2326]中的轨道定义一致
Video Analytics　视频分析算法 用于分析视频数据和产生数据描述的算法或程序
Wi-Fi Protected Access Wi-Fi 授权程序　一套由 Wi-Fi 联盟创建用于保证安全的程序

3.2 缩写

AAC （Advanced Audio Coding） 高级音频编码
ASN （Abstract Syntax Notation） 信息的抽象句法
AVP （Audio/Video Profile） 音视频情景
AVPF （Audio/Video Profile for rtcp Feedback） 实时的音视频情景
BLC （Back Light Compensation ） 背光补偿
BSSID (Basic Service Set Identification)基础服务集鉴定
CA (Certificate Authority) 认证授权
CBC (Cipher-Block Chaining) 加密块链模式
CCMP (Counter mode with Cipher-block chaining Message authentication code Protocol)计数器模式密码块链消息完整码协议
DER (Distinguished Encoding Rules)　可辨别编码规则

DHCP (Dynamic Host Configuration Protocol)　动态主机设置协议
DHT (Define Huffman Table)　定义霍夫曼表
DM (Device Management)　设备管理
DNS (Domain Name Server)　域名服务器
DQT (Define Quantization Table)　定义量化表
DP (Discovery Proxy)　查找服务
DRI (Define Restart Interval) 定义重启间隔
EOI (End Of Image) 图像的结束
FOV (Field Of View)　视场
GW (Gateway) 网关
HTTP (Hypertext Transfer Protocol)　超文本传输协议
HTTPS (Hypertext Transfer Protocol over Secure Socket Layer)超文本传输协议安全
IO (I/O Input/Output) 输入/输出
IP (Internet Protocol)　互联网协议
IPv4 (Internet Protocol Version 4) 互联网协议 4 版
IPv6 (Internet Protocol Version 6) 互联网协议 6 版
Ir (Infrared)　红外线
JFIF (JPEG File Interchange Format) 文件交换格式
JPEG (Joint Photographic Expert Group)联合图像专家组
MPEG-4 (Moving Picture Experts Group – 4) 运动图像专家组 4
MTOM (Message Transmission Optimization Mechanism)消息传输优化机制
NAT （Network Address Translation） 网络地址转换
NFC （Near Field Communication） 近距离无线通讯技术
NTP （Network Time Protocol） 网络时间协议
NVA （Network Video Analytics Device） 网络视频分析器
NVC （Network Video Client） 网络视频客户端
NVD （Network Video Display） 网络视频显示
NVT （Network Video Transmitter） 网络视频发射器
NVS （Network Video Storage Device） 网络视频存储设备
OASIS （Organization for the Advancement of Structured Information Standards）促进信息结构标准进步的组织机构
ONVIF （Open Network Video Interface Forum） 公开的网络视频接口论坛
POSIX （Portable Operating System Interface） 可移植性操作系统接口
PKCS （Public Key Cryptography Standards） 公钥标准
PSK （Pre Shared Key） 预共享密钥
PTZ （Pan/Tilt/Zoom） 云台全方位（上下、左右）移动及镜头变倍、变焦控制
REL （Rights Expression Language） 权限表达语言
RSA （Rivest ,Sharmir and Adleman）公钥加密算法
RTCP （RTP Control Protocol） 实时控制传输协议
RTP （Realtime Transport Protocol） 实时传输协议
RTSP （Real Time Streaming Protocol） 实时流传输协议
SAML （Security Assertion Markup Language） 安全断言标记语言
SDP （Session Description Protocol） 会话描述协议
SHA （Secure Hash Algorithm） 安全散列算法

SOAP （Simple Object Access Protocol） 简单对象访问协议
SOI （Start Of Image） 图像的开始
SOF （Start Of Frame） 侦开始
SOS （Start Of Scan） 扫描开始
SR (Sender Report) 发送报告
SSID (Service Set ID) 服务集标识符
TCP (Transmission Control Protocol) 传输控制协议
TLS (Transport Layer Security) 传输层安全
TKIP (Temporal Key Integrity Protocol) 临时密钥完整性协议
TTL （Time To Live） 生存时间
UDDI （Universal Description, Discovery and Integration） 通用描述、发现与集成服务
UDP （User Datagram Protocol） 用户数据报协议
URI （Uniform Resource Identifier） 统一资源标识符
URN （Uniform Resource Name） 统一资源名称
USB （Universal Serial Bus） 通用串行总线
UTC （Coordinated Universal Time）世界标准时间
UTF （Unicode Transformation Format） Unicode（统一码）转换格式
UUID （Universally Unique Identifier）通用唯一识别码
WDR （Wide Dynamic Range） 宽动态范围
WPA （Wi-Fi Protected Access） Wi-Fi 网络安全存取
WS （Web Services） 网络服务
WSDL （Web Services Description Language） Web 服务描述语言
WS-I （Web Services Interoperability） 网络服务的互通性
XML （eXtensible Markup Language） 可扩展标记语言

4 概述

ONVIF 是基于网络视频的使用案例，适用于局域网和广域网。规范始于一个核心套接口函数配置和通过定义它们的服务类接口实现控制网络视频设备。网络视频设备包括 NVT、NVD、NVS 和 NVA。这个架构在未来还会慢慢扩展和增强。

该框架涵盖不同网络视频环境下的各个阶段，从网络视频设备部署和配置阶段到实时流处理阶段。

这个规范涵盖了设备发现、设备配置、事件、PTZ 控制、视频分析和实时流媒体直播功能，以及搜索，回放和录像录音管理功能。

所有的服务使用同一种的 XML schema（XML 文档的结构），所用到的数据类型都在“ONVIF schema”中定义。不同的服务在各章节和 WSDL 文档中定义。

4.1 Web 服务

web 服务是一种集成应用程序的标准化方法的名称，它是基于 IP 网络，使用了开放的，平台独立的 web 服务标准，如 XML，SOAP1.2 [prat 1] 和 WSDL1.1 ，其中 XML 被用做数据描述的语法，SOAP 用于消息传递，WSDL 用来描述服务。

这个框架是建立在 web 服务标准上的，定义在标准里的所有配置服务都表示为 web 服务操作，并在 WSDL 中定义，使用 HTTP 作为底层的通信机制。

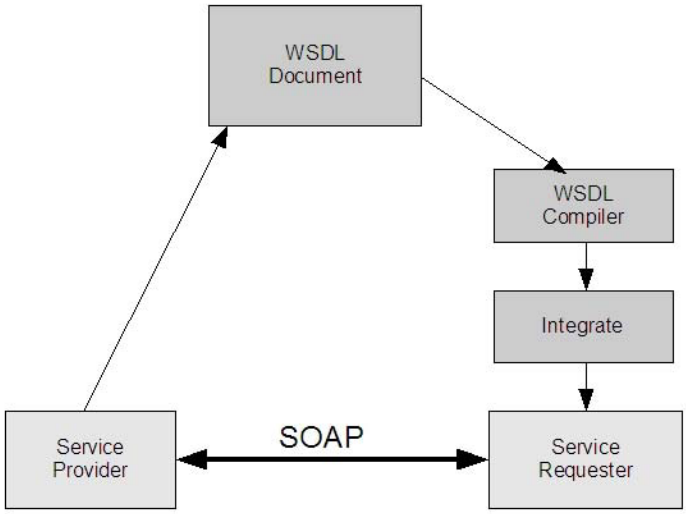


图 1：基于 web 服务的开发原理

图 1 对 Web 服务开发的基本原理进行了一个概括。服务供应者（设备）实现 ONVIF 的服务或者其它服务，这些服务采用基于 XML 的 WSDL 语言进行描述， 然后，由 WSDL 描述的文档将作为服务请求（客户端）实现或者整合的基础。WSDL 编译工具的使用简化了客户端的整合过程，WSDL 编译工具能生成与平台相关的代码，也就是说，客户端开发者可以通过这些代码把 web 服务整合到应用程序中。

Web 服务端和客户端的数据交互采用 SOAP 消息交换协议。SOAP 是一个轻量级的，基于 XML 的消息传递协议，对 Web 服务请求和应答消息进行 SOPA 封装，形成 SOPA 请求和应答消息，然后再传送到网络。SOAP 消息独立于任何的操作系统或协议，而且可以使用各种不同的网络传输协议进行传送。ONVIF 定义了一致的 SOAP 消息传输协议，用于描述 Web 服务。在规范中，

Web 服务概述部分讲解了各种 ONVIF 服务，命令语法，错误处理机制和采用的网络安全机制。

为了确保互操作性，所有定义的服务都遵循 Organization (WS-I) basic profile 2.0 的建议和使用文档/文字封装模型。

4.2 IP 配置

IP 配置部分主要讲 IP 配置需要服从的要求和建议。IP 配置包括：

- IP 网络通信功能
- 静态 IP 配置
- 动态 IP 配置

4.3 设备发现

在本规范中定义的配置接口都是基于 WS-Discovery 标准的 Web 服务接口。该标准的使用，使得重复使用一个合适的现有的 Web 服务发现框架成为可能，而不是需要定义一个全新的服务或者寻址服务定义。

这个标准介绍了一种适用于视频监控目的，具体的发现行为。例如，一个完全可互操作的设备发现，需要一个完整的服务定义和服务搜索标准，为了实现这种方式，规范包含设备类型和范围定义。

一个成功的发现会提供设备服务地址，一旦客户端有了设备的地址，客户端就能通过设备服务接收详细的设备信息，详见 4.5 章节。

除了标准的网络服务发现协议，规范还支持远程发现代理，即使客户端和设备处于不同的管理网络域内，也可以通过远程发现代理找到注册设备。

4.4 设备类型

设备类型体现了一个设备的基本功能。规范定义了以下几种设备类型：

- 网络视频传输设备（NVT）
- 网路视频显示设备(NVD)
- 网路视频存储设备(NVS)
- 网络视频分析设备(NVA)

对于每个设备类型的一些服务是强制性的，详细请参考 5.1.1，一个设备可以支持其他可选的服务，设备通过设备发现机制发布有效的可选服务。

4.5 设备管理

设备管理功能都是通过网络服务来实现。设备服务是设备提供其他所有服务的入口点。设备服务 WSDL 详见设备管理 WSDL 文件。设备管理接口由以下几类组成：

- 功能
- 网络
- 系统
- 安全

4.5.1 功能

客户端可以通过功能类命令获取设备提供的服务，并且确定哪些是设备通用的服务，哪些是备供应商特定的服务。功能由不同的设备服务构成并进一步划分出子类功能如下：

- 分析
- 设备
    - 功能
    - 网络
    - 系统
    - 输入/输出
    - 安全
- 事件
- 成像
- 媒介
- PTZ
- 驱动 IO
- 显示
- 记录
- 查找
- 重放
- 分析设备

对于不同类型的功能，它们的服务命令和参数设置只对特定的服务和子类服务有效。

4.5.2 网络

网络的规范化管理功能命令：

- 读取和设置主机名，
- 读取和配置 DNS，
- 读取和配置 NTP，
- 读取和设置动态的 DNS，
- 读取和配置网络接口，
- 使能/不使能和列出网络协议，
- 读取和配置默认的网关
- 读取和配置 zero
- 读取和设置默认的网关、读取、增加和删除 IP 地址滤波器。

4.5.3 系统

系统命令用于管理以下的设备系统设置：

- 读取设备信息
- 进行系统备份
- 读取和设置系统数据和时间
- 恢复出厂设置
- 固件升级
- 读取系统日志
- 读取设备信息（支持信息）
- 重启
- 读取和设定设备发现参数。

4.5.4 系统信息检索

系统信息，如系统日志，供应商特定支持信息和配置备份图像，能够通过 MTOM 或 HTTP 协议恢检索。

MTOM 方式通过 GetSystemLog, GetSystemSupportInformation 和 GetSystemBackup 命令来实现。HTTP 方式通过 GetSystemUris 命令来实现，文件可以用 HTTP GET 命令重检索的 URIs 中下载。

4.5.5 固件升级

关于固件升级，提供了两种方式。第一种是用 MTOM 通过 UpgradeSystemFirmware 命令发送新的固件映像。

第二种要分为两个阶段进行，第一阶段客户端送 StartFirmwareUpgrade 命令告知设备准备固件升级，然后通过 HTTP POST 发送固件映像。

HTTP 方式用于资源受限设备，该设备在正常工作状态下不能接收新的固件映像。

4.5.6 系统还原

系统还原允许设备从一个备份映像中恢复设备配置信息，系统还原也提供了两种实现方式。第一种通过 MTOM 用 RestoreSystem 命令发送备份映像；第二种用 StartSystemRestore 命令，然后用 HTTP POST 协议发送备份映像。

4.5.7 安全

设备安全管理配置命令：

- 读取和设置使用的安全策略，
- 处理用户凭证和设置，
- 处理 HTTPS 服务证书，
- 使能/不使能 HTTPS 客户身份认证，
- 密钥生成和证书下载功能，
- 处理 IEEE802.1X 客户端证书，

- 处理 IEEE802.1X 认证授权，
- IEEE802.1X 配置

4.6 设备 IO

设备 IO 服务提供指令用于恢复和配置设备的输入输出。

设备 IO 服务支持以下设备接口：

- 视频输出
- 视频源
- 音频输出
- 音频源
- 中继输出

现有接口命令表：

- GetVideoOutputs - 读取设备所有的视频输出
- GetVideoSources -读取设备所有的视频源
- GetAudioOutputs -读取设备所有的音频输出
- GetAudioSources -读取设备所有的音频源
- GetRelayOutputs -读取设备所有的中继输出

视频输出，视频源，音频输出和音频源支持的命令：

- 设置设备名称配置（Set<device name>Configuration）- 改变某个接口的配置
- 读取设备名称配置（Get< device name >Configuration）- 读取某个接口的配置
- 读取设备名称配置选项（Get< device name >ConfigurationOptions）- 读取某个接口的属性值。

中继输出支持的命令：

- SetRelayOutputSettings - 改变一个中继输出的配置
- SetRelayOutputState - 设置逻辑状态

设备 IO 服务的 WSDL 在[DeviceIOService.wsdl]中有详细说明。

4.7 图像配置

图像服务提供了图像属性的配置和控制服务，WSDL 作为架构一部分，详见图像 WSDL 文件。

- 图像配置包括以下操作：
- 读取和配置成像参数（如曝光时间，增益和白平衡）
- 读成像配置选项（对于成像参数的范围）
- 变焦
- 停止正在进行的调焦
- 读取读取位置和焦距状态

4.8 媒体配置

媒体配置通过媒体服务来处理。媒体配置用于决定在规范中定义的流媒体属性，设备通过媒体服务提供媒体配置。媒体服务 WSDL 详见媒体 WSDL 文件。

4.8.1 媒体配置文件

实时视频流和音频流配置通过媒体配置文件控制，一个媒体配置文件管理一个音视频源到一个音视频编码器、PTZ 和分析配置。根据不同的功能，NVT 呈现出不同的配置文件（配置文件可以动态的改变）。

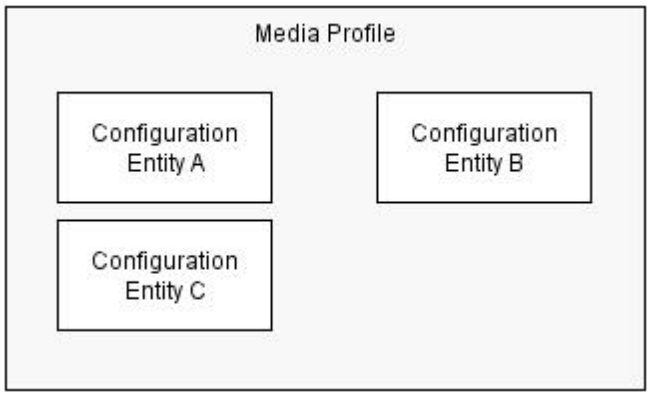


图 2：一个媒体配置剖析

一个具有媒体配置服务的设备在启动后至少提供一个媒体配置文件。一个设备应该提供一些，现成的，最常见的媒体配置文件以供使用。

配置文件含有一个“固定”属性，该属性表示用于表示配置文件能否被删除，这个属性可以通过 NVT 定义。

一个配置文件由一系列相关联的配置实体构成。通过 NVT 能够创建静态或动态的配置，例如，根据现有编码资源，NVT 能够创建一个动态的配置。一个配置实体是以下的其中之一：

- 视频源配置
- 音频源配置
- 视频编码器配置
- 音频编码器配置
- PTZ 配置
- 视频分析配置
- 元数据配置
- 音频输出配置
- 音频解码器配置

一个配置文件由全部的或部分的配置实体组成，NVT 的功能决定了一个特定的的配置实体能否成为配置文件的一部分。例如，只有在支持音频的设备中，其配置文件才能包含音频源和音频编码两个配置实体。

一个完整的配置文件举例，如图 3：

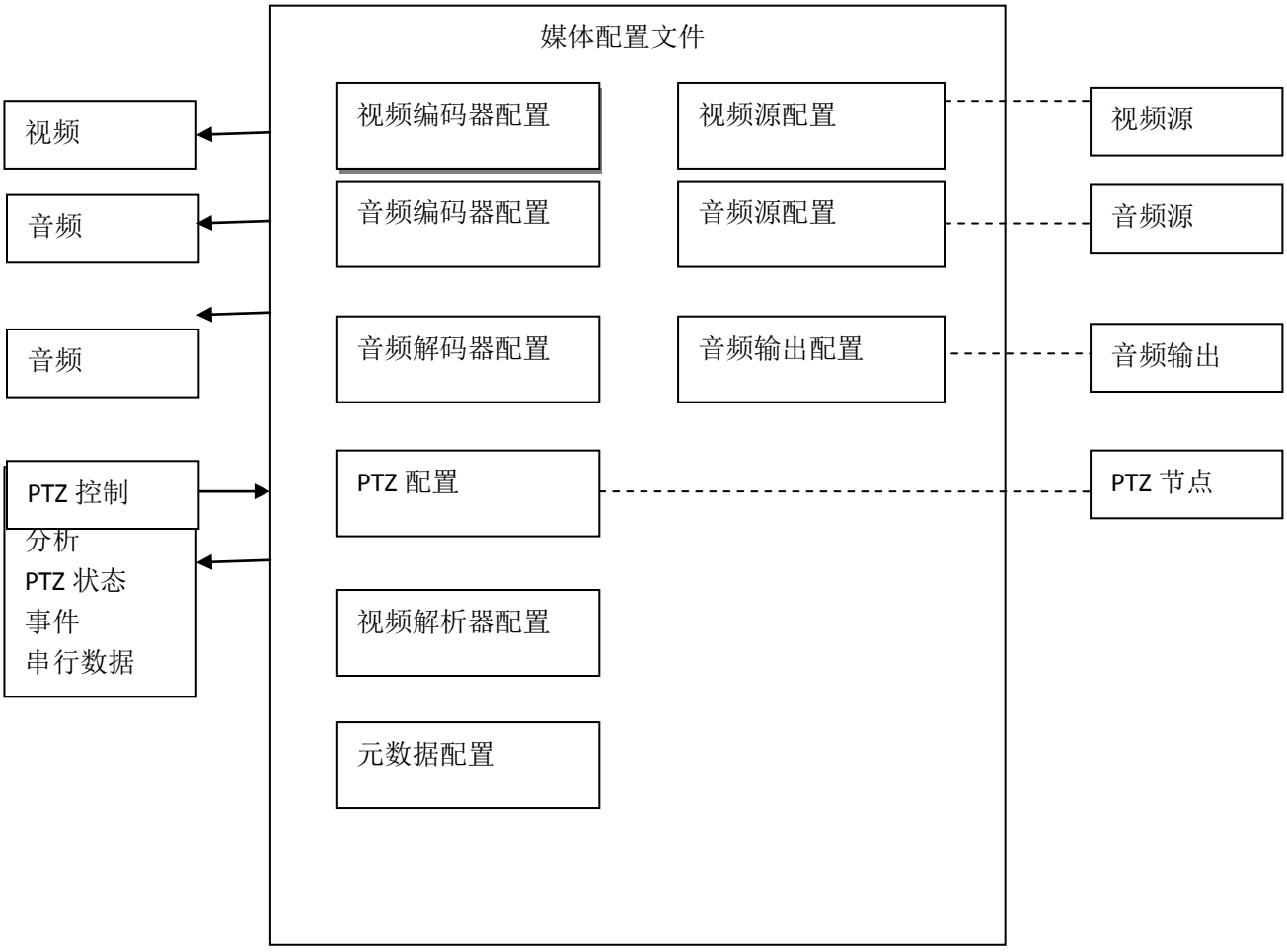


图 3：完整的配置文件

一个媒体配置文件描述的是，在媒体流中如何给予和给予什么给客户端，同时怎样处理 PTZ 的输入和分析。

读取源列表命令：

- GetVideoSources －在设备上读取所有存在的视频源
- GetAudioSources －在设备上读取所有存在的音频源
- GetAudioOutputs －在设配上读取所有存在的音频输出

管理媒体配置文件命令：

- CreateProfile －创建一个新的媒体配置文件
- GetProfiles －读取所有存在的媒体配置文件
- GetProfile －读取某个媒体配置文件
- DeleteProfile －删除某个媒体配置文件
- Add<configuration entity> －增加某个配置实体到媒体文件
- Remove<configuration entity> －从配置文件中删除某个配置实体

管理配置实体命令：

- Get<configuration entity>Options －读取某个配置实体的当前属性值
- Set<configuration entity> －设置一个配置实体
- Get<configuration entity>s －读取某类的所有配置实体

- Get<configuration entity> – 读取某个配置实体
- GetCompatible<configuration entity>s – 读取某个媒体配置文件所有的配置实体

配置实体<configuration entity>是指配置实体的类，例如一个用于读取一个视频编码器配置的完整命令：

GetVideoEncoderConfiguration

初始化和操作音视频流命令：

- GetStreamUri – 为某个媒体请求一个有效地 RTSP 或 HTTP 流标识符
- StartMulticastStreaming – 利用某个配置文件开始多路出送
- StopMulticastStreaming – 停止多路传输
- SetSynchronizationPoint – 在正在进行的流传输中插入一个同步节点
- GetSnapshotUri – 为某个能获取 JPEG 快照的配置文件请求一个有效地 HTTP 标识符

举例说明一个配置文件是如何在一个客户端运行实现的，请详见第五章。

4.9 实时流

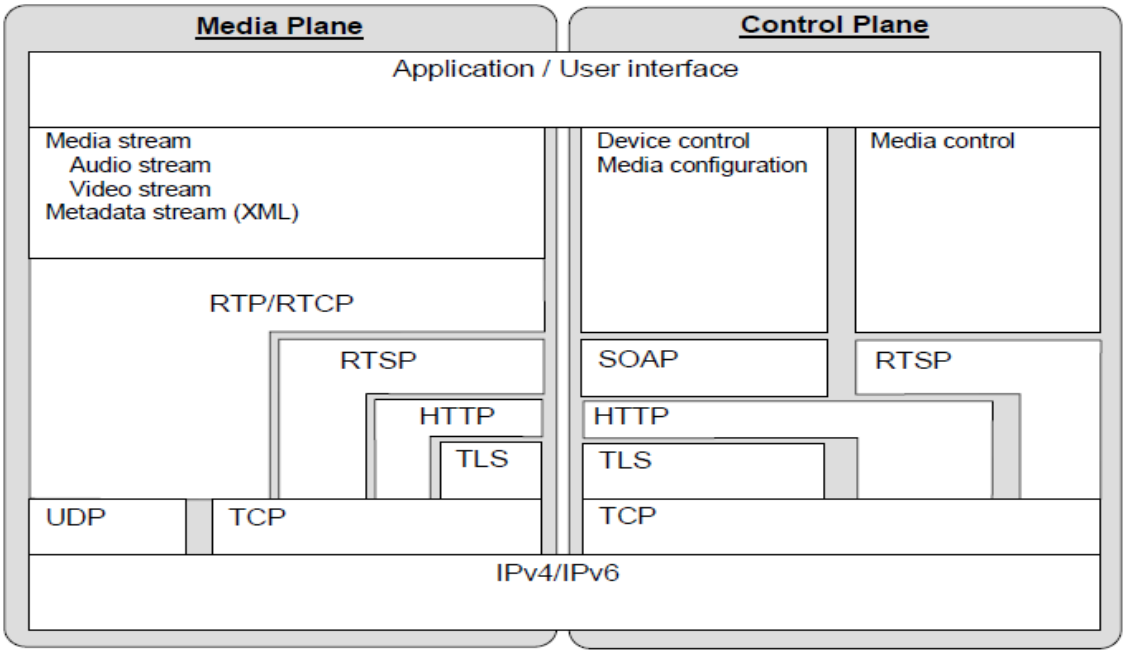


Figure 4: Layer structure

本标准规定了流媒体的选项和格式。首先要区分媒体平面和控制平面，如图 4 所示。为了提供可互操作性的媒体流服务，媒体流（音视频、元数据）服务选项都是使用 RTP 协议进行描述的。

元数据流容器格式支持定义良好的，实时流分析，PTZ 状态和消息数据。

媒体配置是通过 SOAP/HTTP 协议完成的，在 4.6 章会详细的讨论媒体配置服务。

媒体控制通过在 RFC 2326 中定义的 RTSP 协议完成，这个标准利用了 RTP，RTCP 和 RTSP 协议分析，以及基于 RTP 扩展的 JPEG 和组播控制机制。

这个标准介绍了基于 RTSP 标准的扩展，用于允许双向的流连接。

流配置支持的视频编解码：

- JPEG(通过 RTP),见 12.1.3
- MPEG-4 简单级(SP) [ISO 14496-2]
- MPEG-4 高级简单级(ASP) [ISO 14496-2]
- H.264，基准[ISO 14496-10]
- H.264，要点[ISO 14496-10]
- H.264，扩展[ISO 14496-10]
- H.264，高级的[ISO 14496-10]

支持的音频编解码：

- G.711 [ITU-T G.711]
- G.726 [ITU-T G.726]
- AAC [ISO 14496-3]

4.10 事件处理

事件处理是基于 OASIS WS-BaseNotification 和 WS-Topics 规范，这些规范可以重用丰富的通知架构，而不需要重新定义处理原则、基本格式和通信方式。

按照 WS-BaseNitification 协议，通过 pullpoint 通知模式可实现防火墙穿越，然而这个模式不支持实时通知，因此，这个规范定义了一个可供选择的 pullpoint 通知模式和服务接口，这种模式在使用 WS-BaseNotification 架构时，允许客户端在防火墙后接收实时通知。

一个完全标准的事件需要规范的通知。然而通知的主题在很大程度上依赖于应用需求，这个规范定义了一系列基本的通知主题，建议设备支持这些通知主题，详细请参阅附录 A。此外，对于一些服务，这个规范扩展了基本的，含有强制事件的通知主题。

事件及相关扩展服务的 WSDL 在事件 WSDL 文件中进行详细的说明。

4.11 PTZ 控制

PTZ 服务用于控制视频编码设备的云台全方位（上下、左右）移动及镜头变倍、变焦控制。PTZ 服务的 WSDL 应用详见 PTZ WSDL 文件。

PTZ 控制原则遵循媒体配置模式（见 4.8），主要由三部分组成：

- PTZ Node －用于管理 PTZ 设备和功能的低级 PTZ 实体
- PTZ Configuration －保存某个 PTZ 节点的 PTZ 配置
- PTZ Control Operation －PTZ,预设和状态操作

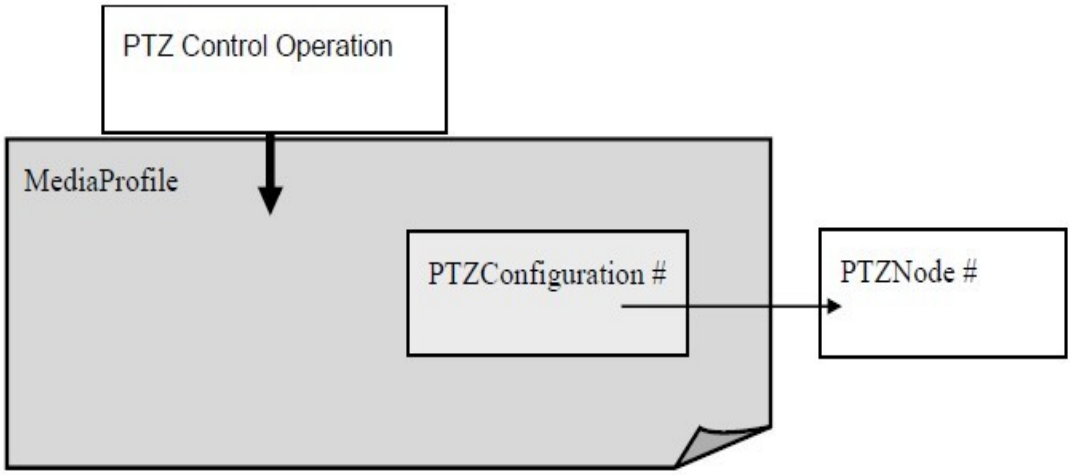


Figure 5: PTZ control model

一个具有 PTZ 功能的 NVT 可能有一个或许多个 PTZ 节点，一个 PTZ 节点可以是一个机械 PTZ 设备驱动，也可以是一个上传到视频编码器上的 PTZ 驱动或是一个数字的 PTZ 设备驱动。PTZ 节点是 PTZ 控制中最低级的实体，它指定支持的 PTZ 功能。

PTZ 配置在每个媒体配置文件中设置，并通过以下配置命令实现：

- 读取和配置摄像机转动，倾斜，变焦
- 读取摄像机转动，倾斜，变焦配置参数

该标准定义了以下的 PTZ 控制操作：

- PTZ 完全的，相对的，连续的动作操作
- 停止操作
- 读取 PTZ 状态信息（如位置，错误和移动状态）
- 读取，设置，删除和移动预设位置
- 读取，设置和移动到中心位置

4.12 视频分析

视频分析应用被分为图像分析和具体应用两部分。这两部分之间的接口产生一个抽象概念，即可根据出现的对象来描述一个场景，视频分析应用简化为对场景描述的比较和场景规则（如被禁止通过的虚拟线或定义的一个多边形保护区域），其他的规则可以是表示内部物体的行为，如物体紧跟另一个物体（对追尾的侦测），这些规则也可以用来禁止某些物体行为，如限速。

如图 6 所示：视频分析应用对应的两部分又可简称为视频分析引擎和规则引擎，再加上事件和行为部分，构成了一个视频分析框架。

视频分析架构由元素和接口组成，每个元素提供了一个功能，在一个完整的视频分析解决方案中，对应到一个语义上独特的实体。接口是单向性的，并且是具有特定内容的信息实体。对于规范来说，只有接口是受支配的。架构的核心能力就是分配任何元素或邻近元素组到网络中的任何设备。

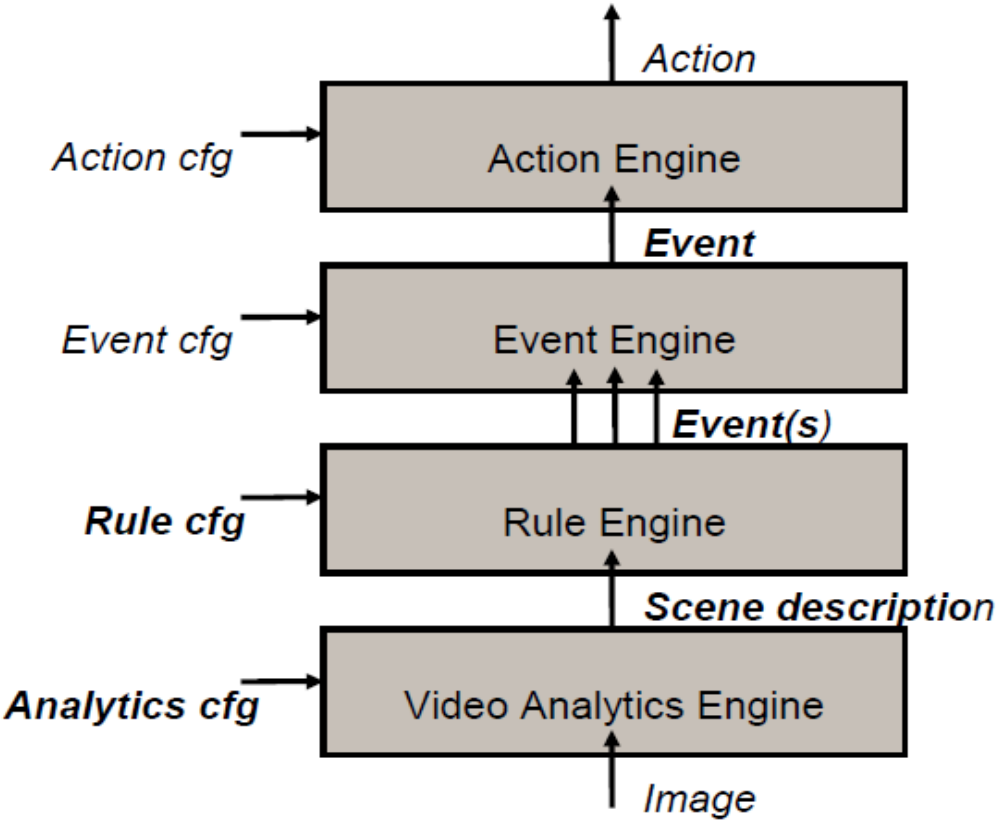


Figure 6: Video analytics architecture

在规范中定义了以下接口：

- 分析配置接口
- 场景描述接口
- 规则配置接口
- 事件接口

规范定义了一个视频分析引擎的配置框架。这个架构使设备能按照客户端的配置进行分析，这样的模块配置能够通过客户端动态的增加，删除或修改。假如设备支持的话，允许客户端并行地运行多个视频分析模块。

视频分析引擎的输出被称作场景描述。场景描述是真实场景基于对象的抽象，不管场景是静态的还是动态的，它们都可看作真实场景的一部分。本规范定义了一个基于 XML 的场景描述接口，它包括数据类型和数据传输机制。

规则描述了怎样解读场景和怎样处理这方面的信息。规范定义了标准的规则语法和方式，用于应用程序和设备之间的交流。

一个事件表示场景描述分析的状态和相关连的规则，事件接口包括事件引擎元素的输入和输出，事件接口通过一般的通知和主题架构来处理（见 4.10）.

视频分析服务的 WSDL 是 onvif 框架的一部分，在分析的 WSDL 文件中描述。

4.13 分析设备

分析设备服务必须用于独立的分析设备，用于执行对媒体数据流或元数据增强媒体流的评估过程，在同一时间，评估可能包含多个媒体流或元数据增强媒体流。

分析设备服务是从现场或储存设备中接收媒体流或元数据增强数据流。假如分析器正在分析未压缩数据，那说明设备具有解码功能。

分析设备服务是由客户端操作的，用于配置独立分析设备的属性和功能。独立分析设备不具备反向通道功能。

使用事件服务可以获得分析设备服务的输出，此外，分析设备服务还支持 GetStreamUri 命令。

4.14 显示

显示服务用于客户端控制和配置显示设备。服务提供了窗格显示的功能,就是每个窗格占用物理显示（实际显示设备屏幕）的一定区域。窗格显示可以配置多个音频输入，输出与视频输出之间的映射关系，配置也引用了一个接收对象，即接收显示数据的对象。除此之外，显示服务还支持显示窗格属性的检索和配置。

layout 定义了窗格在显示器的布局情况 (例如单个显示或分成 4 块显示)。服务提供了获取面板显示和改变显示布局命令。和显示布局命令一样，服务还介绍了获取视频输出的编解码信息的命令。

4.15 接收器

一个接收器是就是一个 RTSP 客户终端。接收器被用于其他处理媒体流的设备服务中，如显示、记录和分析设备服务。一个接收器对应一个配置，配置决定 RTSP 终端连接到哪里和使用那些连接参数。

一个接收器能够运行在三种不同的模式下：

总是连接 - 接收器尝试保持一个持续的连接到设置端点

断开连接 - 接收器不尝试连接

自动连接 - 当媒体流在发出请求连接时连接

一个单一的接收器可以被多个用户使用。例如,为了录制并分析一个流数据，那么录制工作作和分析引擎就要使用同一接收器。假如接收端使用“自动连接”模式,那么只要记录或解析引擎是工作的，接收器都会处于连接状态,只有它们都不工作时才会断开。

在接收服务中，通过调用 CreateReceiver 和 DeleteReceiver 操作，可以手动的创建和删除接收器，或在其他服务中，也可自动的创建和删除接收器。例如,假如创建一个带有“AutoCreateReceiver”属性的录制工作,那么它就会自动创建一个接收器，并将录制工作附加到这个接收器上，删除记录工作的同时也会删除接收器。

4.15.1 同步点

因为接收器利用 RTSP 地址来确定流数据的来源,所以它们并不需要使用传输设备的 web 服务接口，也就意味着它们不能使用 SetSynchronizationPoint 命令，具体请参考 11.18.1。取而代之的是,接收器应该用 PLI 消息来请求一个同步点，PLI 消息详见[RFC 4585]。

4.16 存储

标准提供了一系列的接口，用于支持可互操作的网络存储设备，例如网络存储器（NVR），数字视频接收器（DVR）和嵌入式存储摄像机。

存储服务支持以下功能：

- 记录控制
- 查找
- 回放

这些功能通过三个相关联的服务来提供。

记录服务使客户端能管理记录和配置数据源到记录器的数据传输。管理记录包括记录的创建和删除。

搜索服务是使客户端能查找存储器中的记录信息，例如，首先生成一个可见时间列表方式的记录视图，接着通过搜索包含在元数据跟踪记录中的事件，在一组记录中查找我们感兴趣的数据。

回放服务使客户端能回放记录的数据，包括音频、视频和元数据。回放服务提供媒体流开始，停止播放，以及改变播放速度和回放方向的功能，也允许客户端从存储设备中下载数据，支持设备数据导出功能。

4.16.1 存储模式

在本标准的存储接口提出了一个数据在存储设备上的逻辑视图，该视图与数据可能在磁盘上的真实存储方式完全无关。

在存储模式中关键的概念是记录（recording），在规范中术语记录代表一个容器，容器中是一整套相关联的音视频和元数据轨道（tracks），通常这些数据来自于同一数据源，例如摄像机。一条记录可以容纳任何数量的轨道，一条轨道可看作是一个可以无限长的时间轴，在某个时间段保存的数据。

一条记录最少能够容纳三种类型的轨道，分别是音频，视频，元数据。在一些实现的记录服务的中，每种类型的轨道又可以包含多条轨道，例如，同一个记录能够容纳两个视频轨道，一个包含低分辨率或低帧率流和一个包含高分辨率或高帧率流。

Figure 7: Storage Model with Tracks

值得注意的是，存储接口不能说明设备的内部存储结构，特别是，一个记录并不代表是在磁盘上的一个单一的文件，而在很多存储设备中，一个记录存储了一系列的文件。比如，一些摄像机通过创建不同的文件来记录每一个发生的警报，尽管每一个文件可以作为不同的记录来描述，但是在标准中这种模式的目的就是为了把他们聚合在一条记录中。

在一条记录内数据记录的区域用一对事件来表示，每对事件都由记录开始时间的事件和记录结束时间的事件构成。通过使用查找服务的 FindRecordings 和 FindEvents 方式，客户端可以构建记录的逻辑画面。

假如元数据被记录，元数据轨道就能保存所有数据源产生的事件（请参考事件处理和元数据配置相关章节）。另外，设备也能够记录 ONVIF 定义的历史事件（请参考查找服务中的记录事件描述），这里包括像记录数据范围的开始和结束的信息，设备也能够记录关于供应商的一些历史事件。设备产生的事件不会被插入到已存在的元数据记录轨道中。查找服务中的 FindEvents 模式能帮助找到所有的记录事件。

4.16.2 记录

记录服务使客户端能管理记录和配置数据源到记录的数据传输。管理记录包括记录和轨道的创建和删除。

记录工作是从一个记录源传送数据到一个记录。一个记录源可以是一个接收服务创建的接收对象，也可以是一个在本地设备上编码数据的媒体文件。媒体文件可以用作一个在嵌入式存储摄像机的源。

存储数据到一条记录，一个客户端首先要创建一条记录和确保记录有必要的轨道。然后客户端创建一个记录工作，这个工作能从一个或多个源获取数据并存储到记录的轨道中。

客户端可以创建多个记录工作，并把所有工作的数据都记录到同一条记录中，假如多个记录工作都处于活动状态，设备就会根据优先权原理在记录中定义的轨道之间选择。客户端可以在任何时候改变记录工作的模式，因此需要提供实现报警记录或人工记录特征的方法。

尽管接收对象由 ReceiverTokens 鉴定，但是为了从其它设备接收数据，记录工作依赖于接收服务。

4.16.3 查找

查找服务使客户端能够查找在存储器设备中的记录信息，例如，首先生成一个可见时间列表方式的记录视图，接着通过搜索包含在元数据跟踪记录中的事件，在一组记录中查找我们感兴趣的数据。

查找服务提供以下功能：

- 查找每个 recordings 的记录和信息
- 在元数据和历史数据中查找事件
- 在元数据中查找 PTZ 位置
- 在元数据中查找信息如从 EPOS（电子销售点）系统中

实际的搜索通过查找和结果获取两个操作实现，并且是异步进行的。每一个查找操作启动一个搜索对话，客户端就可以从查找对话中用增量的方式获取搜索结果，或者一次获取所以结果，这依赖于具体的实现和搜索的范围。有四对查找操作分别对应记录，记录事件，PTZ 位置和元数据.

FindRecordings and GetRecordingSearchResults

FindEvents and GetEventSearchResults

FindPTZPosition and GetPTZPositionSearchResults

FindMetadata and GetMetadataSearchResults

4.16.4 回放

回放服务提供了一种机制，用于对存储的视频、音频和元数据进行回放，这个机制也可以用来从存储设备中下载数据以便提供导出功能。

回放服务是基于 RTSP[RFC 2326]协议的,然而因为 RTSP 不能直接地支持所有的回放需求，所以加入了一些协议的扩展。尤其是一个 RTP 头扩展被定义允许一个绝对的时间戳与每个接入单元（视频帧）相关联，以及传输信息的流连续性。

在回放服务中 GetReplayUri 命令返回记录的 RTSP URL 来允许它用 RTSP 来回放。

4.17 安全

这个条款描述了网络视频安全需求，规范定义了两个不同等级的安全机制：

传输级安全

消息级安全

规范也定义了如下的基于端口的网络安全：

IEEE 802.1X

一般安全的要求，定义和传输安全要求在 22 章有详细的说明。消息级安全要求在 5.12 有详细的说明。IEEE 802.1X 要求在 8.4.7 安全管理章节有详细的说明，通过设备管理服务实现安全管理，详细请参考 4.5.7。

5 Web 服务框架

所有管理和配置命令都是基于 web 服务的。

标准的目的是：

- 设备（NVT，NVD，NVS，NVA）是服务提供者
- 客户端（client）是服务请求者

一个典型的网络视频系统可以由多个客户端来处理一台设备的所有配置和管理业务，但是，控制和网络视频接收功能之间可能存在区别。提供服务的设备，也可以作为客户端，未来的规范可能会在系统中引入其它的实体和接口。

Web 服务还需要一个共同的方式来发现服务提供者，这个可以通过使用 UUDI 实现，即“统一描述、发现和集成协议”([UDDI API ver2], [UDDI Data Structure ver2])。UDDI 协议使用了服务代理来发现服务，因为本规范的目的是设备，而 UDDI 模型不是以设备为导向的，所以 UDDI 和服务代理不在本协议之内规定。

根据本协议规定，设备（服务提供者）使用基于[WS-Discovery]技术的 WS 发现机制。设备发现的原理在第 7 章描述。

web 服务允许开发者自由地定义服务和消息交换机制，这可能会引起互操作性的问题。Web 服务互操作性组织(WS-I)开发了标准的配置文件和指引，以创建可互操作的 Web 服务。设备和客服端应该遵守在[WS-I BP 2.0]定义的指引。本协议的服务描述遵循[WS-I BP 2.0]的建议。

5.1 服务概述

一个遵循 ONVIF 的设备需要支持许多规范定义的 web 服务。例如在本规范定义的 web 服务：

- 设备服务
- 媒体服务
- 事件服务

设备服务是符合 ONVIF 标准的设备的目标服务，并且是所有其他设备服务的入口点。

设备管理的入口点绑定在：

http://onvif_host/onvif/device_service

5.1.1 服务要求

一个设备应提供设备管理和事件服务。根据设备的功能，设备可以支持任何其它的服务。根据设备类型（NVT, NVD, NVS, NVA）的不同，必须为它们追加额外的服务。确切的合规性要求看做是本规范中不同服务定义的一部分。

如果设备支持某一种服务，那么设备就应该能够响应所有命令（相应的 WSDL 服务命令）。假如服务不需要某一命令并且设备不支持这个命令，那么设备就应该响应一个错误码：

env:Receiver,

ter:ActionNotSupported

错误码的定义，详见 5.11.2。

表一：设备类型的服务请求

	NVT	NVS	NVD	NVA
Device	M	M	M	M
Event	M	M	M	M
Media	M			
PTZ	C			
Imaging				
Analytics				M
Recording Control		C		
Recording Search		M		
Replay Control		M		
Device IO	M		M	
Receiver		C	M	M
Display			M	
Analytics Device				M

表一显示了对于不同类型的设备要求哪些服务。其中“M”代表强制性的的服务（Mandatory），一个强制性的服务被一些相关特性的设备支持的话用“C”来标记。

5.2 WSDL 概述

WSDL 是一种的 XML 格式文档，它把 web 服务描述成一组能够操作面向文档或者是面向过程信息的端点。这些操作和消息被描述得很抽象，然后绑定到一个具体的网络协议和消息格式来定义一个端点。相关的具体端点结合成抽象的端点（服务）。WSDL 可以扩展到描述端点和端点的信息，而不管信息的格式或者使用何种网络协议通信。

这个规范遵循 WSDL1.1 的规范，并使用了文档的规范模式。

WSDL 文档由以下几部分组成：

- 类型 ： 定义数据类型，使用 XML 格式定义
- 消息 ：定义输入和输出消息的内容
- 操作 ：如何定义输入和输出信息相关的逻辑操作。
- 端口类型 ： 多个操作集合在一起
- 绑定 ： 一个特定端口类型用于交换信息的协议规范。
- 端口 ： 指定绑定的端口地址
- 服务： 通常是指一组相关的端口集合

5.3 命名空间

在本标准前中使用的前缀和命名空间列于表 1。这些前缀不是标准的一部分，标准可以用任何前缀实现

表二：本规范定义命名空间

前缀	命名空间 URI	描述此规范中为 XML 格式描述
tt	http://www.onvif.org/ver10/device/wsdl	WSDL 设备服务的名字空间
tds	http://www.onvif.org/ver10/media/wsdl	WSDL 媒体服务名字空间
trt	http://www.onvif.org/ver20/imaging/wsdl	WSDL 成像服务的名字空间
timg	http://www.onvif.org/ver10/events/wsdl	WSDL 事件服务的名字空间
tev	http://www.onvif.org/ver20/ptz/wsdl	PTZ 控制服务的名字空间
tptz	http://www.onvif.org/ver20/analytics/wsdl	分析服务的名字空间
tan	http://www.onvif.org/ver10/error	ONVIF 定义错误的名字空间
ter	http://www.onvif.org/ver10/network/wsdl/	规范中用于远程设备发现服务的名字空间
dn	http://www.onvif.org/ver10/topics	ONVIF 主题名字空间
tns1	http://www.onvif.org/ver10/analyticsdevice/wsdl	WSDL 分析设备服务的名字空间
tad	http://www.onvif.org/ver10/deviceIO/wsdl	
tmd	http://www.onvif.org/ ver10/display/wsdl	ONVIF 主题设备名字空间
tls	http://www.onvif.org/ ver10/display/wsdl	WSDL 显示设备服务的名字空间
trv	http://www.onvif.org/ ver10/receiver/wsdl	WSDL 接收端服务的名字空间
trc	http://www.onvif.org/ ver10/recording/wsdl	WSDL 记录服务的名字空间
trp	http://www.onvif.org/ ver10/replay/wsdl	WSDL 重播服务的名字空间

tse	http://www.onvif.org/ ver10/search/wsdl	WSDL 搜寻服务的名字空间

表三：引用的命名空间（前缀）

前缀	URI 名字空间	描述
wsdl	http://schemas.xmlsoap.org/wsdl/	WSDL 网络框架的命名空间
wsoap12	http://schemas.xmlsoap.org/wsdl/soap12/	WSDL SOAP 1.2 的绑定名字空间
http	http://schemas.xmlsoap.org/wsdl/http/	WSDL HTTPGET & POST binding 的名字空间
soapenc	http://www.w3.org/2003/05/soap-encoding	SOAP 1.2 [SOAP 1.2, Part 2]定义的编码空间
soapenv	http://www.w3.org/2003/05/soap-envelope	SOAP 1.2 [SOAP 1.2, Part 1]定义的信封命名空间
xs	http://www.w3.org/2001/XMLSchema	XS [XML-Schema, Part1] and[XMLSchema,Part 2]定义的信封命名空间
xsi	http://www.w3.org/2001/XMLSchema-instance	XML 格式的信封命名空间
d	http://schemas.xmlsoap.org/ws/2005/04/discovery	定义在[WS-Discovery].中的设备发现命名空间
wsadis	http://schemas.xmlsoap.org/ws/2004/08/addressing	在 [WSDiscovery]. 提及的设备寻址空间
wsa	http://www.w3.org/2005/08/addressing	定义在[WS-Addressing]. 中的设备寻址空间
wstop	http://docs.oasis-open.org/wsn/t-1	[WSTopics]规范的格式命名空间
wsnt	http://docs.oasis-open.org/wsn/b-2	[WSBaseNotification]空间的格式命名空间
xop	http://www.w3.org/2004/08/xop/include	XML-binary Optimized 包装优化的命名空间

此外，本标准里提及没有前缀的命名空间列于表三。

表四：引用的命名空间（没有前缀）

Namespace URI	Description
http://docs.oasis-open.org/wsn/t-1/TopicExpression/Concrete	Topic expression dialect defined for topic expressions.
http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet	The ONVIF dialect for the topic expressions.
http://www.onvif.org/ver10/tev/messageContentFilter/ItemFilter	The ONVIF filter dialect used for message content filtering.
http://www.onvif.org/ver10/tptz/ZoomSpaces/PositionGenericSpace	The ONVIF standard zoom position space for PTZ control.
http://www.onvif.org/ver10/tptz/PanTiltSpaces/PositionGenericSpace	The ONVIF standard pan/tilt position space for PTZ control.
http://www.onvif.org/ver10/tptz/ZoomSpaces/TranslationGenericSpace	The ONVIF standard zoom translation space for PTZ control.
http://www.onvif.org/ver10/tptz/PanTiltSpaces/TranslationGenericSpace	The ONVIF standard pan/tilt translation space for PTZ control.
http://www.onvif.org/ver10/tptz/ZoomSpaces/VelocityGenericSpace	The ONVIF standard zoom velocity space for PTZ control.
http://www.onvif.org/ver10/tptz/PanTiltSpaces/VelocityGenericSpace	The ONVIF standard pan/tilt velocity space for PTZ control.
http://www.onvif.org/ver10/tptz/ZoomSpaces/SpeedGenericSpace	The ONVIF standard zoom speed space for PTZ control.
http://www.onvif.org/ver10/tptz/PanTiltSpaces/SpeedGenericSpace	The ONVIF standard pan/tilt speed space for PTZ control.

5.4 类型

数据类型在 XML Schema 描述的第一和第二部分中定义。此规范中所有定义的数据类型都包括在[ONVIF Schema]中，并且能够从下网址下载。

http://www.onvif.org/onvif/ver10/schema/onvif.xsd

5.5 消息

根据 WSDL1.1 可知，操作使用基于 XML 格式的输入和输出消息描述，消息部分包含了信息内容。

在这个规范中，一个消息应包括两个主要的要素：

- 消息的名称
- 消息的部件

消息名称指定元素的名字，并且这个名字在 WSDL 文档中的操作定义会被使用。消息的名称确定了这个消息的具体名字。

WSDL 的消息部件要素用来定义消息的实际格式。尽管在 WSDL 定义消息可能有多个部件，但这个规范遵循了[WS-I BP2.0]规定，不允许多个部件元素出现在一个消息中。因此我们一直用同一个名称“parameters”作为消息部件的名字。此标准中，以下的 WSDL 符号是用于消息的：

<message name="'Operation_Name'Request">
<part name="parameters" element="'prefix':'Operation_Name'"/>
</message>
respective,
<message name="'Operation_Name'Response">
<part name="parameters" element="'prefix':'Operation_Name'Response"/>
</message>

这里的“prefix”是命名空间的前缀，消息是在该名空间中定义的。

在这个规范中使用消息的具体类型，这些类型封装了多个部件，允许消息中含有多个参数或者数据。

5.6 操作

操作是定义在 WSDL 端口类型声明中的。操作可能是下面两种类型之一：

- 单向 ：服务提供者接收消息。
- 请求-应答（双向） ： 服务提供者接收到消息并发出相应的消息。

根据操作，可以使用不同的端口类型。
操作名称就确定了此操作的名字
在此规范中操作使用下表五中的列出的信息来定义：

表五：本规范所用的操作描述

操作名字	操作类型
信息名字	描述
操作名字的请求	请求信息的描述 Typer1 Namer1 [ar1][br1] Typer2 Namer2 [ar2][br2] Typern Namern [arn][brn]
操作名字的应答	应答信息的描述 Types1 Names1 [as1][bs2] Types2 Names2 [as2][bs2] : Typesn Namesn [asn][bsn]
错误信息的名字	在操作具体故障已经定义的情况下，这一领域介绍了错误消息的结构定义

误码	描述
Code Subcode Subcode	指定的错误操作码的描述

表 5 描述栏包含一个元素列表，这个列表在请求消息和响应消息中是分开表示的。括号里的值表示具体类型的元素允许出现的次数的上限和下，例如，Names2 出现了至少 as2 次，最多 bs2 次。

大多数命令并没有定义任何具体错误消息，如果一个消息定义了，那么将出现在应答消息之后，并它遵从上表的约束。

5.6.1 单向操作

当服务提供者收到一个控制消息，并且还没有发出任何确认的消息时，就用到了单向操作类型。此规范中，单向操作仅仅是用来实现设备发现和事件。

此操作类型由单一的输入消息来定义的。

用下表来描述单向的操作类型：

Operation_Name	One-way
Message name	**Description**
'Operation_Name'Request	*Description of the request message.* *Type*$_1$ **Name**$_1$ [a_1][b_1] *Type*$_2$ **Name**$_2$ [a_2][b_2] : *Type*$_n$ **Name**$_n$ [a_n][b_n]

本表对应 WSDL 符号：

```
<operation name="'Operation_Name'">
<input message="'prefix':'Operation_Name'"/>
```

5.6.2 要求-应答操作类型

当服务提供者收到一个消息，并以相应的消息应答的时候，就用到请求—应答操作类型了。这种类型是通过一个消息输入，一个消息输出，多个故障信息来定义的。使用如下表来描述要求—应答操作类型

Operation_Name	Request-Response
Message name	**Description**
'Operation_Name'Request	*Description of the request message.* $Type_{r1}$ **$Name_{r1}$** $[a_{r1}][b_{r1}]$ $Type_{r2}$ **$Name_{r2}$** $[a_{r2}][b_{r2}]$: $Type_{rn}$ **$Name_{rn}$** $[a_{rn}][b_{rn}]$
'Operation_Name'Response	*Description of the response message.* $Type_{s1}$ **$Name_{s1}$** $[a_{s1}][b_{s2}]$ $Type_{s2}$ **$Name_{s2}$** $[a_{s2}][b_{s2}]$: $Type_{sn}$ **$Name_{sn}$** $[a_{sn}][b_{sn}]$
'"FaultMessage_Name'	*In the case that operation specific faults are defined, this field describes the structure of the defined fault message.*

Fault codes	**Description**
Code *Subcode* *Subcode*	*Description of the operation specific fault.*

此表相应的 WSDL 符号：

<operation name="'Operation_Name'">
<input message="'prefix':'Operation_Name'"/>
<output message="'prefix':'Operation_Name'Response"/>
<fault name> = "Fault" message = "'prefix':'FaultMessage_Name'">

5.7 端口类型

一个端口的类型是一组有名字的抽象操作的和抽象信息。一个单独端口的类型是由不同的操作组合起来的。

在这个规范中所有的操作名称都被归了种类，每个操作类包含了一个或者多个操作，但每个种类只有一种操作类型，并且被整合成了单一的端口类型。一个单向操作和一个请求—应答操作绝不可能出现在相同的端口类型中去。

5.8 绑定

绑定为一个特定的端口类型定义了具体协议和数据传输格式规范。对于一个给定的端口类型，都有可能包含任何数量的绑定。

端口类型是先前定义的类型并且"绑定"是一个以大写字母的开始字符串，它定义了绑定的名称。

在本标准中，设备绑定的定义应遵循[WS-I BP 2.0]中的要求，这意味着 WSDL SOAP 1.2 绑定将被使用。SOAP 绑定可以有不同的方式，一个设备在操作层面上应该使用"文档"的绑定方式。

对于各自的服务，它们的绑定均定义在 WSDL 规范中

5.9 端口

对于一个绑定，各自的终端被一个唯一的地址来指定。每个端口都应该赋予一个独一无二得名字。端口定义包含一个名称和一个绑定属性
本规范不强制任何端口命名规则

5.10 服务

服务是一组相关的端口。这个规范不强制任何服务的命名规则。

5.11 错误处理

就像其他任何协议一样，错误可能在通信，协议和信息加工的时候产生。
该规范错误处理可以处理以下类别的错误：

- 协议错误
- SOAP 的错误
- 应用程序的错误

5.11.1 协议错误

协议错误是指错误协议消息，(它可能包含非法的头值，也有可能在非期望时间或经历了一个套接字超时的时候被接收了)。为了说明和解释协议错误，HTTP 和 RTSP 协议已经定义了一套标准的状态码[例如，1XX，2XX,3XX,4XX,5XX]。通过此标准，当收到故障报告的时候，设备和客户端将采用合适的 RTSP 和 HTTP 协议定义状态码来报告错误

5.11.2 SOAP错误

SOAP 错误一般是由于 web 服务操作错误或者是在 SOAP 消息处理期间导致的。所有这些 SOAP 错误需要使用 SOAP 错误消息机制报告和处理。通过 SOAP 错误机制，SOAP 规范提供了一个定义良好的框架来处理错误。

一个 SOAP 错误信息是在消息实体中带有一知名元素（soapenv：Fault）的 SOAP 信息。为了更详细地理解错误，SOAP 定义了 SOAP 错误信息结构，该结构含有各种成分：

- 错误码
- 子码
- 原因
- 节点和角色
- 错误细节

子码和错误细节是用来携带应用的具体的错误信息。
本标准对具体的错误使用了一个单独的名字空间

ter = “http://www.onvif.org/ver10/error”.

对于不同的 web 服务，SOAP 错误信息被看作是各子 web 服务定义中的一部分。就像本规范规定的那样，服务器和客户端应当使用 SOAP1.2 错误信息处理机制，并且要遵循 WS - I 基本

简介 2.0 错误处理建议。
如下的例子就是故障信息：

```
HTTP/1.1 500 Internal Server Error
CONTENT-LENGTH: bytes in body
CONTENT-TYPE: application/soap+xml; charset=”utf-8”
DATE: when response was generated
<?xml version=”1.0” ?>
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soapenvelope"
xmlns:ter="http://www.onvif.org/ver10/error"
xmlns:xs="http://www.w3.org/2000/10/XMLSchema">
<soapenv:Body>
<soapenv:Fault>
<soapenv:Code>
<soapenv:Value>fault code </soapenv:Value>
<soapenv:Subcode>
<soapenv:Value>ter:fault subcode</soapenv:Value>
<soapenv:Subcode>
<soapenv:Value>ter:fault subcode</soapenv:Value>
</soapenv:Subcode>
</soapenv:Subcode>
</soapenv:Code>
<soapenv:Reason>
<soapenv:Text xml:lang="en">fault reason</soapenv:Text>
</soapenv:Reason>
<soapenv:Node>http://www.w3.org/2003/05/soapenvelope/
node/ultimateReceiver</soapenv:Node>
<soapenv:Role>http://www.w3.org/2003/05/soapenvelope/
role/ultimateReceiver</soapenv:Role>
<soapenv:Detail>
<soapenv:Text>fault detail</soapenv:Text>
</soapenv:Detail>
</soapenv:Fault>
</soapenv:Body>
</soapenv:Envelope>
```

表总结了一般的 SOAP 错误代码（错误码定义在 SOAP 1.2 第 1 部分：消息框架中）服务器和客户端可以定义应用程序使用的，额外的错误子码。

我们区分一般错误和具体的错误。任何命令都能产生一般错误信息，具体的错误涉及到一个特定的命令或命令集。特定命令的具体错误在命令的定义表中说明。

在下面的表格，错误码，子码和错误原因是一些标准值。说明列添加其它信息。

5.11.2.1 常见的故障

表五列出了常见的错误码和字码，所有的服务器和客户端必须实现对如下表所列的错误进行处理，任何 web 服务的命令会返回一个或者几个通用的错误码。

列出的无子码错误没有任何子码值

表六：常见的故障

Fault Code	Subcode	Fault Reason	Description
env:VersionMismatch		SOAP version mismatch	The device found an invalid element information item instead of the expected *Envelope* element information item.
env:MustUnderstand		SOAP header blocks not understood	One or more mandatory SOAP header blocks were not understood.
env:DataEncodingUnknown		Unsupported SOAP data encoding	SOAP header block or SOAP body child element information item is scoped with data encoding that is not supported by the device.
env:Sender	ter:WellFormed	Well-formed Error	XML Well-formed violation occurred.
env:Sender	ter:TagMismatch	Tag Mismatch	There was a tag name or namespace mismatch.
env:Sender	ter:Tag	No Tag	XML element tag was missing.
env:Sender	ter:Namespace	Namespace Error	SOAP Namespace error occurred.
env:Sender	ter:MissingAttr	Required Attribute not present	There was a missing required attribute.
env:Sender	ter:ProhibAttr	Prohibited Attribute	A prohibited attribute was present.
env:Sender	ter:InvalidArgs	Invalid Args	An error due to any of the following: • missing argument • too many arguments • arguments are of the wrong data type.
env:Sender	ter:InvalidArgVal	Argument Value Invalid	The argument value is invalid.
env:Sender	ter:UnknownAction	Unknown Action	An unknown action is specified.

env:Sender	ter:OperationProhibited	Operation not Permitted	The requested operation is not permitted by the device.
env:Sender	ter:NotAuthorized	Sender not Authorized	The action requested requires authorization and the sender is not authorized.
env:Receiver	ter:ActionNotSupported	Optional Action Not Implemented	The requested action is optional and is not implemented by the device.
env:Receiver	ter:Action	Action Failed	The requested SOAP action failed.
env:Receiver	ter:OutofMemory	Out of Memory	The device does not have sufficient memory to complete the action.
env:Receiver	ter:CriticalError	Critical Error	The device has encountered an error condition which it cannot recover by itself and needs reset or power cycle.

5.11.2.2 具体的错误

具体的错误，仅适用于特定的命令或命令集。具体的错误被声明为服务的一部分定义在这个标准里面。

5.11.2.3 HTTP 错误

如果服务器开始等待入栈的消息并且没有接收到 SOAP 消息，服务器不会产生一个常见的 SOAP 故障，相反会发送一个 HTTP 错误作为回应。

表七：HTTP 错误

HTTP Error	HTTP Error Code	HTTP Reason
Malformed Request	400	Bad Request
Requires Authorization	401	Unauthorized
HTTP Method is neither POST or GET	405	Method Not Allowed
Unsupported message encapsulation method	415	Unsupported media

服务器应该避免将其内部错误报告出来，因为这可能会成为一个安全隐患，可能被外界所利用。

5.12 安全

定义在该标准中的服务应该确保使用 WS 安全框架。WS 安全规范定义了一套可以用来提供服务消息完整性和保密性标准的 SOAP 的扩展。这个框架允许多个不同的使用令牌的安全模块。以下是目前已经定义了的令牌：

- 用户名令牌简介[WS-UsernameToken]
- X.509 安全令牌简介[WS-X.509 TOKEN]
- SAML 令牌简介[WS -SAML TOKEN]
- KERBEROS 令牌简介[WS -KERBEROS TOKEN]
- 权利表达语言令牌简介[WS - REL TOKEN]

服务器和客户端将支持用户名令牌就像在 WS 安全和 5.12.2 中指定的那样。也可能支持 WS 安全规范定义的其它配置文件。

用户名令牌文件仅仅是给出了一个基本层面的安全。在一个安全性很重要的系统里面，建议把设备接入方式配置成是基于 TLS 安全技术的。这样的话，用户名令牌消息层面的安全就与 TLS，客户端，服务器身份验证有机结合起来了，这样就保护了传输级的安全性，在许多系统中就得到一个可以接受的安全水平了。

NVIF 兼容设备在 RTSP 和 HTTP 验证时，使用与 web 服务一样的证书。对于使用令牌配置文件的用户，在 RTSP 和 HTTP 验证中，还要使用摘要式身份验证[RFC 2617]。

ONVIF 兼容设备应该能在 WS 层上验证 WS 请求。HTTP 在这只是用来作一种传输协议，并且设备不能在此层上验证 WS 请求。

ONVIF 兼容设备应该能在 RTSP 层上验证 RTSP 请求，如果 HTTP 是用于隧道协议，那么在此水平上，RTSP 请求不应该验证。

在认证 RTSP 和 HTTP 的方法的时候，ONVIF 兼容设备将使用和 WS 部分一样的用户证书。对于使用令牌配置文件的用户，在 RTSP 和 HTTP 验证中，还要使用摘要式身份验证[RFC 2617]。

5.12.1 基于用户访问控制

在 SOAP 的消息层面上，WS 安全架构允许保护和认证。对于 ONVIF 设备来说，这些认证机制是用来为 onvif 服务建立一个访问安全策略。这个规范允许安全策略架构基于四个不同的用户层：

1．管理员
2．操作员
3．媒体使用户
4．匿名用户

使用这些分类可以定义不同用户的详细接入策略。未经授权的用户被划分到匿名类并且设备不允许用户被添加到匿名用户的类别。

设备的用户或者系统管理员可以明确的定义访问安全策略。策略配置文件的确切格式在本规范之外。

获取和设置访问安全策略的命令被定义在 8.4 章节 。

5.12.2 用户令牌配置文件

WS 安全规范强制需要实现令牌配置只有是用户令牌配置[WS-UsernameToken]。客户端应该使用在[WS-UsernameToken]定义的随机数和时间戳，且服务器应当拒绝任何不使用随机数和时间戳的用户名令牌。

本标准定了一套命令来管理用户名称的令牌文件证书。这这些命令允许用户结合不同的

用户级别使用，用户级别定义在 5.12.1 中。

5.12.2.1 密码推导

使用几个相同证书的设备会在不经意间引进了一些安全风险。然而要求用户为每个设备提供单独的证书是不可取的，客户端应实施以下密码算法的推导：

UA 表示任意一个用户（arbitrary user）。P_UA 表示用户 UA 接入系统中设备要使用的密码。此外，NEP 表示系统中一个特定设备的设备终端服务点的参考值。最后，PE_UA 表示客户端用于访问某个特定的设备的密码，客户端应该按照如下方法来计算 PE_UA 的密码：

PE_UA = base64(HMAC_SHA-1(UA + P_UA,NEP+“ONVIF password”))

出现”+”的地方表示连续，这里“ONVIFpassword”是一个 ASCLL 字符串，该字符串不包含字符长度或者 null 结尾。

例如下面的十六进制值：4F 4E 56 49 46 20 70 61 73 73 77 6F 72 64。

HMAC_SHA-1 是作为一种基本算法，一种被定义在[RFC 2104] using SHA-1 [FIPS 180-2]中的算法。用于 HMAC 功能中的关键值就是用户密码，P_UA 直接映射为等值的二进制数。类似地，在传输 PE_UA 值到设备之前，它的值应该用 ASCLL 码表示。

BASE64 被描述在[RFC3548]中，注意 BASE64 的操作结果是就是实际密码，并且将使用这个密码。

5.12.2.1.1 例子

假设以下是密码，由客户端（ASCLL）：“用户”和“VRxuNzpqrX”组成

UA = 75 73 65 72

P_UA = 56 52 78 75 4E 7A 70 71 72 58

接下来，假设设备具有如下设备服务终端的参考值：

Urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6

那么由此产生的密码将被计算为：

PE_UA = base64(HMAC_SHA-1(P_UA,NEP+”ONVIF password”)) =

base64(HMAC_SHA-1(75736572565278754E7A70717258,

F81D4fAE7DEC11D0A76500A0C91E6BF6+4F4E5649462070617373776F7264)) =

base64(16 E5 C5 A9 4D DE 8A 97 6D D7 2F 55 78 5F C2 D0 6B DA 53 4A)=

FuXFqU3eipdt1y9VeF/C0GvaU0o=

那么由此产生“FuXFqU3eipdt1y9VeF/C0GvaU0o=”这个密码，在一个特定的设备中将被客户端用来配置用户凭据，并且也可以被客户端用来接入设备。

6 IP 配置

设备和客户端通过开放或封闭的 IP 网络进行通信。本标准对使用那种网络类型没有任何限制和要求。然而，根据在标准 4 中指定的体系架构建立实体之间的通信链路应该是可行的。设备的 IP 配置包括多个参数，如 IP 地址和默认网关。

设备应至少有一个网络接口，可以接入 IP 网络，同样，客户端也应该至少有一个网络接口可以接入 IP 网络，允许数据在设备和客户端之间传输。该设备和客户端应该支持 IP V4 协议，也应该支持 IP V6 协议。

通过网络或者本地配置接口可以完成设备的静态 IP 配置。

根据[RFC3927]规范，设备应支持本地链路地址的动态 IP 配置。根据[RFC4862 规范]，一个支持 IP6 的设备应该支持无状态的 IP 配置，并且能支持邻居发现（ [RFC4861]规范）。

根据[RFC2131]规范，设备应该支持动态的 IP 配置。根据[RFC3315]规范，支持 IPV6 的设备可以支持有状态的 IP 配置。

该设备可以支持一些额外的配置机制。

设备的网络配置可以通过一个 IP 网络接口或本地接口（诸如 USB 接口，串行接口，蓝牙接口，或者 NFC）。通过一个本地的接口的 IP 配置是在本规范以外。通过指定在 8.2 章节的参数配置接口应能配置设备的 IP 配置。通过这个网络配置接口规范，一个设备用户可以启用或禁用任何 IP 地址的配置选项。默认的设备配置将使 DHCP 和动态本地链接地址配置直接启用。即使设备通过一个静态地址配置，它也应该有本地连接的默认地址。

当设备链接到 IP V4 网络，应该要考虑地址分配优先权（本地链路与路由的地址），就像[rfc3927]中推荐的那样。

更多说明 IP 网络如何连接实现的具体方法不在的本标准之内。

7 设备发现

7.1 概述

客户端可以使用动态服务发现协议 [WS-Discovery]来发现可用的设备，一个兼容于此规范的设备应该实现在[WS-Discovery]中指定的目标服务和客户端两个角色。

发现代理的角色就像在[WS-Discovery]里面描述的那样，不应该被设备或客户端支持。（本规范会在 7.4 节中介绍一种可选的发现代理角色）。如在[WS-Discovery]的第 3 节描述那样，实现了客户端角色的设备将忽略与发现代理的交互方式。相反，本规范定义了一个新的允许远程发发现的发现代理角色。远程发现的实现需要存在一个发现代理和一个系统提供者，该提供者在系统中愿意提供远程发现服务，并实现了在第 7.4 节中定义的发现代理角色。

[WS-Discovery]描述了 UUID，参考 2.6 节，建议端点使用 URI 格式的 UUID，但在本协议中改写了该建议，相反，采用了[RFC4122]中的 URN 格式，即统一资源命名。（详细请看 7.3.1 节）

7.2 操作模式

设备可以有两种操作模式：

- 可发现的
- 不可发现的

根据[WS-Discovery]规范，设备处于可发现模式时，一旦接入网络，就组播“hello”消

息，或者发送状态改变消息。此外，它始终监听探测和解析消息，并发送相应的响应。一个设备处于不可发现模式时，不会监听，也不会回应这些信息。

设备的默认模式是可发现模式，为了阻止拒绝服务攻击，可以通过在 8.3.19 定义的操作来将设备设置成不可发现模式。

7.3 发现定义

7.3.1 终端参考

一个设备或一个充当客户端角色的端点，使用一个 URN:UUID [RFC4122] 作为其端点引用的地址属性。

一个设备或一个充当客户端角色的端点，应该使用一个稳定的，全局唯一标识符作为其端点引用的地址属性，并且在跨网络接口的情况下是不变的。Wsadi：address 和 wsadis：referenceproperties 结合起来提供了一个固定的全球唯一标识符。

7.3.2 服务地址

包含在 Hello 消息中的<d:XAddrs>元素为设备服务地址或设备地址。

该设备应提供一个 80 端口作为设备服务入口点，以便允许防火墙穿越。

7.3.3 Hello

7.3.3.1 类型

设备应该包含设备管理服务的端口类型，比如 tds:Device，在<d:Types>中声明的。

考虑到向后兼容的原因，ONVIF 兼容设备还应当包括在<d:Types>中声明的 dn:NetworkVideoTransmitter。

下面的例子显示了在 SOAP “hello” 实体中的端口类型是如何编码的。

<d:Types>tds:Device</d:Types>.

该消息可能包括其他类型。

7.3.3.2 范围

在 Hello 消息中，设备还应包括<d:Scopes>属性范围。

该设备范围是通过使用[RFC3986]的 URIs 来设置的。本规范定义范围属性如下：

方案属性: ONVIF

权威属性：www.onvif.org

这意味着，所有 ONVIF 定义范围的 URI 都有以下格式：

onvif://www.onvif.org/<path>

该设备可能有其他范围的 URIs。这些 URI 是没有限制在 ONVIF 规定范围中。

表八定义了一个设备的基本能力和其他属性。除了这些标准参数以外，还可以设置设备所有者定义的任何范围参数。通过定义在 8.3 节中的命令，可以列出和设置范围参数。未来版本的规范可能会引入更多的标准化范围参数。

设备还可能有其他范围的 URI。这些 URI 是没有限制的在 ONVIF 规定范围中。

表八 ： 范围参数

类目	定义值	说明

类型	视频编码器	一个视频编码器暗示了这个设备是具有视频编码设备，一个带有网络视频编码的设备应该将网络视频编码器的型号包括在其内。
	云台	一个云台范围暗示了这个设备是一个具有云台设备带有云台支持的设备包括一个值在其范围内的范围输入
	视频解析	一个视频解析的范围暗示了这个设备支持视频解析就像 17 部分的那样，一个带有视频解析支持的设备应该包括一个值在其范围清单内的范围输入
	音频编码器	一个音频编码器范围说明这个设备是一个具有音频编码的设备一个带有音频编码支持的设备应该包括一个值在其范围清单内的范围输入
	网络视频发射机	一个网络视频发射机的范围暗示了这个设备是否是一个 NVT 兼容设备，一个 NVT 应该包括一个其值在其范围清单内的范围输入
	网络视频解码器	一个网络视频的显示范围暗示了这个设备是否是一个 NVD 兼容设备，一个 NVD 应该包括一个值在其范围清单内的范围输入
	网络视频的存储	一个网络视频的存储范围暗示了这个设备是否是一个 NVS 兼容设备，一个 NVS 应该包括一个其值在其范围清单内的范围输入
	网络视频的解析	一个网络视频的解析范围暗示了这个设备是否是一个 NVA 兼容设备，一个 NVA 应该包括一个值在其范围清单内的范围输入
方位	任何字符串和路由值	方位为这个设备定义了一个物理方位，位置值可能是字符串为这个设备描述的物理装置，一个设备应该至少包括一个方位。
硬件	任何字符串和路由值	一个字符串或路径描述装置的硬件，一个设备应该包括至少一个硬件入口在其范围内
名字	任何字符串和路由值	搜索设备的名称，一个设备应该包括至少一个名字入口在它的范围清单内

在范围清单的位置、硬件和名称类别中，一个设备在各个类别中至少包含一个该类型的条目。一个设备可能包含在范围清单内，其他任何额外的范围属性。

在其范围清单内，一个设备可以包括任意个范围，这意味着，例如，一个单元可以定义多个不同位置的范围。“探头”匹配所有范围清单。

7.3.3.2.1 例子

下面的例子说明了范围值的使用方法，这仅仅是个例子，并不表示这些范围参数的类型就是 NVT 配置中的一部分。在这个例子中，我们假设 NVT 用以下的范围来配置：

onvif://www.onvif.org/type/Network_Video_Transmitter

onvif://www.onvif.org/type/video_encoder

onvif://www.onvif.org/type/ptz
onvif://www.onvif.org/type/audio_encoder
onvif://www.onvif.org/type/video_analytics
onvif://www.onvif.org/hardware/D1-566
onvif://www.onvif.org/location/country/china
onvif://www.onvif.org/location/city/bejing
onvif://www.onvif.org/location/building/headquarter
onvif://www.onvif.org/location/floor/R5
onvif://www.onvif.org/name/ARV-453

一个探测设备的客户端使用以下范围的：

onvif://www.onvif.org 可以找到一个匹配的。

类似地,

具有下范围的设备探头：

onvif://www.onvif.org/location/country/china 可以给出一个匹配

具有下范围的设备探头探头：

onvif://www.onvif.org/hardware/D1，无法给出一个匹配

7.3.3.3 地址

设备在“hello”消息中可能包含带有设备服务地址的<d:XAddrs>元素。在 5.12.2.1.1 中为设备定义了 IP 地址配置的规则。

7.3.4 探头和探头匹配

设备匹配探头类型，范围和地址定义，看 7.3.3 HELLO 章节。

设备将至少支持 http://schemas.xmlsoap.org/ws/2005/04/discovery/rfc3986 范围中的匹配规则。此范围匹配的定义与[WS-Discovery]中的范围定义略有不同，因为[RFC2396]由 [RFC 3986]代替。

设备在探头匹配的消息中应该包含带有设备服务地址的<d:XAddrs>元素，这个<d:XAddrs>元素在大多数情况下只包含一个地址来管理和配置接口 ，就像 5.1 规定的那样。

7.3.5 解决和解决匹配

这个规范要求终端地址信息被包含到“HOLLO”信息和探头匹配信息中去，在大多数情况下，没有必要解决匹配和匹配的交换问题.然而，为了与[WS-Discovery]规范能兼容，一个设备应该能处理解决匹配的回应。

7.3.6 BYE

就像在 WS-Discovery 中描述的那样，当设备准备离开网络的时候，设备应该发出一个单向的 BYE 消息.

7.3.7 SOAP错误信息

如果存在一个多播数据包错误，该设备和客户端必须悄悄的丢弃，并且忽略请求。如果所有设备对同一个请求都发送错误响应信息，有可能造成数据包风暴，所以建议不发送错误

响应信息。考虑到完整性，单播数据包错误处理描述在下面。

如果一个设备接收单播探头消息，而且它不支持匹配规则，那么此探头可能不会发送探头匹配消息，相反会产生一个 SOAP 错误（遵循 SOAP 1.2 标准），就像如下所示的那样：

[action] http://schemas.xmlsoap.org/ws/2005/04/discovery/fault

[Code] s12:Sender

[Subcode] d:MatchingRuleNotSupported

[Reason] E.g., the matching rule specified is not supported

[Detail] <d: SupportedMatchingRules>

</d: SupportedMatchingRules>

在扩展或应用中引起的所有错误要遵守 SOAP1.2 中的错误消息规范。在传送一个 SOAP 故障信息给发送者后，应该通知应用程序已经发生错误了。

7.4 远程发现扩展

本节描述的发现扩展需要覆盖更复杂的网络情景。ONVIF 兼容终端并不要求支持这些扩展。，一个支持远程服务发现的设备应当支持本节中定义的发现扩展机制。

此规范中，在本节中定义的远程发现扩展机制可以与本规范定义的基于 WS-Discovery 普通组播方法结合起来使用。比如远程发现扩展机制可以与普通的本地发现一起并行工作。

7.4.1 网络情景

如果客户端和设备不在同一管理区域，客户端使用多播探头去找到和连接设备是不可能的。例如，如果设备或客户端驻留在网络防火墙或 NAT（网关）后面，这就不可能连接到一多播探头。因此需要使用其他的方法，此规范使用了四种不同的网络情景：

1. 该设备驻留在一个私有管理域中，尔客户端在公共网络中，如图 8。
2. 该设备在公共网络中，客户端在一个私有管理域，如图 9。
3. 该设备在一个私有管理域中，客户端在另一个私有管理域中，如图 10。
4. 这设备和客户都位于公共网络中，如图 11。

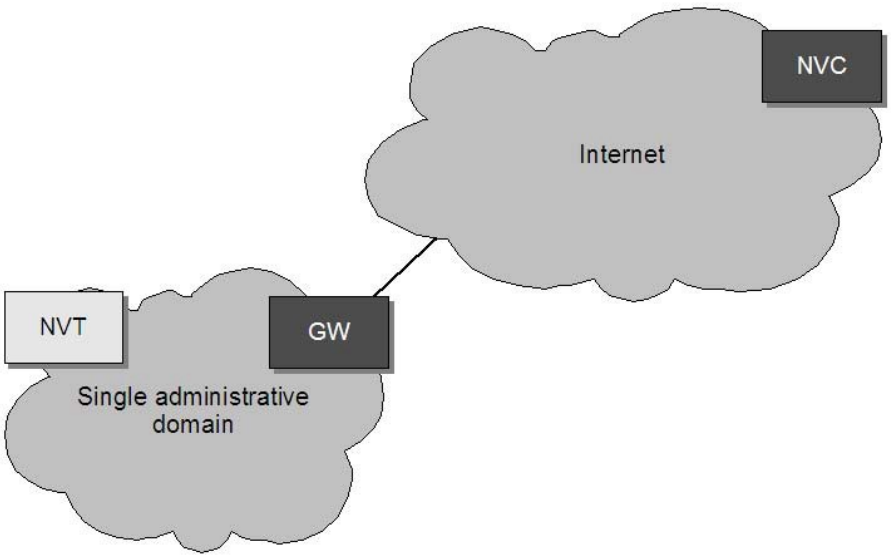


图 8：一个设备，例如一个 NVT, 在一个管理域（私人）和客户端（NVT）在公共网络

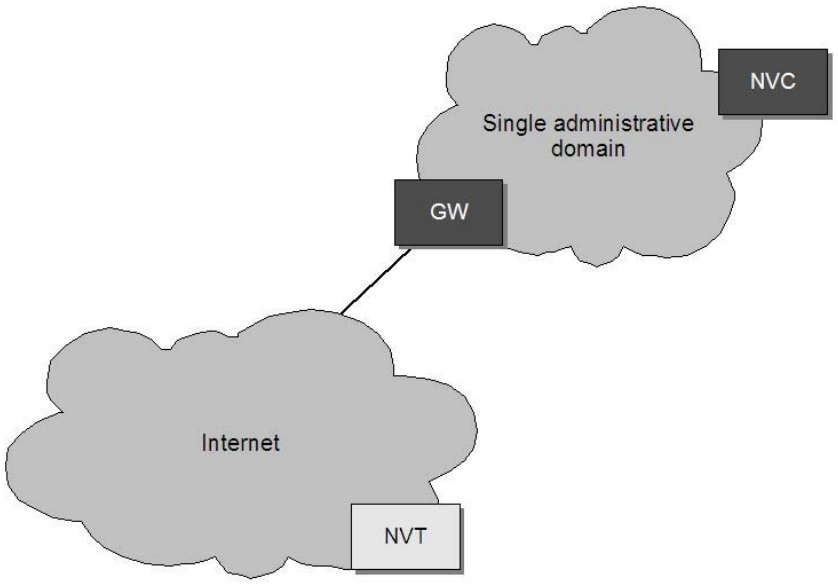


图 9：一个设备，例如 NVT 在公共网络和客户端（NVT）的管理域（私人）

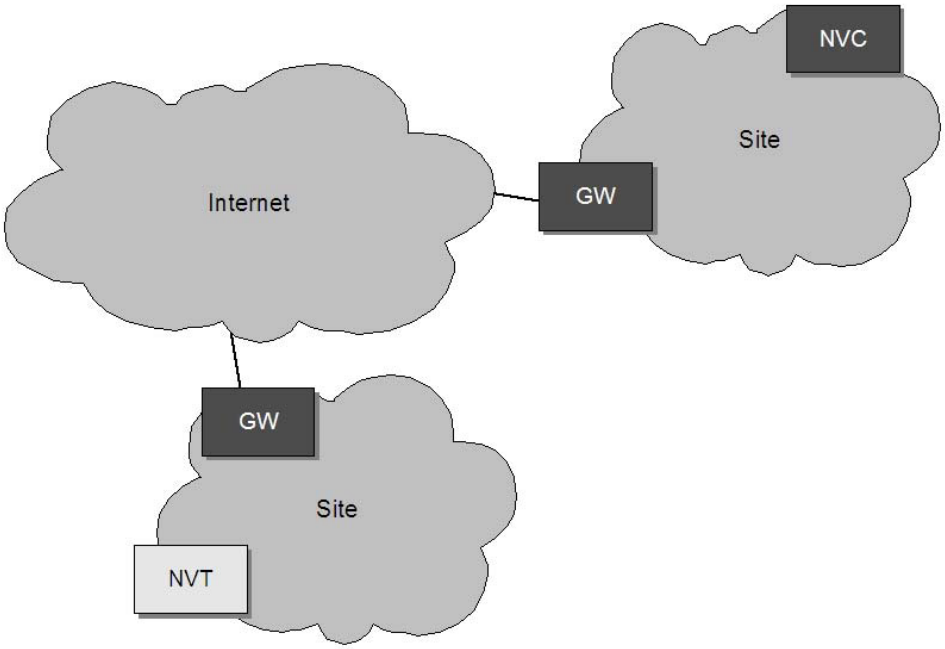


图 10：一个 NVT 设备，在一个管理域（私人）和客户端（NVT）在另一个管理域（私人）

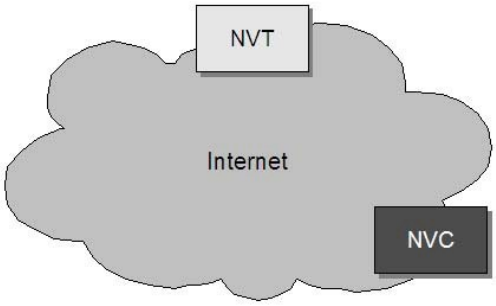


图 11：一个设备，例如 NVT，和客户端（NVT）在公共网络

这个[WS-Discovery]规范介绍了一个发现代理 (DP)来解决其中的一些情景。但是[WS-Discovery]规范并不支持所有在这介绍的网络情景。这个规范定义了一个对于更多复杂的网络情景都可以即插即用的 DP,但这个 DP 规范不兼容于[WS-Discovery]规范。

7.4.2 发现代理

网络管理员为跨越几个管理域的 NVT 广域网配置网络参数，需要在系统中引进一个 DP 终端。DP 执行将下面的任务：

1．听取设备 HELLO 消息并且响应，这些部分定义 7.4.3

2．负责代表注册设备响应来自客户端的探测检查。

这个 DP 可以与设备处在一个管理域中。为了支持这样的网络情景，即客户和设备处在

不同的域中，并且它们之间没有组播连接关系，这时可以把 DP 放到一个公开的网络中，使得设备和客户端都能访问它。设备应当可以找到它的本地 DP 的网络地址，并通过发送“hello”消息到本地 DP，以宣告自己存在。根据本规范，本地 DP 网络地址可以由以下方式获得：

1．直接的地址配置

2．使用域名服务记录查找来发现 DP

设备在接入网络或者本地 DP 网络地址改变的时候，将尝试使用上述任一方式去连接本地 DP（发现代理）。

还可以启用/禁用的设备远程发现注册。支持远程发现的设备应该实现在这定义在 8.3.21 中定义的远程“HELLO”禁用/启用操作。

没有配置本地 DP 地址的设或禁止远程“HELLO”消息的设备不会发送一个远程 HELLO 这定义在 7.4.3。

7.4.2.1 直接的 DP 地址配置

本规范在 8.3.22 节和 8.3.23 节中，介绍了通过网络接口配置本地 DP 地址的设备管理命令。

支持远程发现的设备也可能提供在本地配置“本地 DP”的方式。这些配置是通过设备本地接口完成的，例如设备的串口或者 usb 接口。这些本地配置的方法不在本规范定义。

7.4.2.2 域名服务记录的查找

如果一个设备启用了远程发现，但没有远程 DP 地址配置，它将试着用 DNS 来查找“本地 DP”地址。应使用以下的记录名称和协议定义[rfc2782]：

_onvifdiscover._tcp

为了避使用 DNS SVR 查找 DP，设备在启用远程发现之前，先使用直接地址配置方式，将 DP 地址配置好。

为了使设备用 DP 成功的查询到其他设备，管理员需要将 DP 地址，端口，优先级这些信息记录到使用 SRVS 的 DSN 服务器中。需要有一个或者几个登记服务器，确切的数量由系统负载确定，这些在本协议规范之外。

7.4.3 远程hello和探头行为

定义在[WS-Discovery]中的本地发现模式对远程发现情景不起作用，如果这个设备在防火墙后面，像图 8 和图 10 中的情景那样。如果设备不返回一个公共的网络地址，来自 DP 的单播探头不会自动找到该设备。此外，如果设备在一个防火墙后面，跟随单播匹配探头的设备不可能返回到发现代理中去。该规范针对远程发现定义了一个稍微不同的通信模式来解决这个问题。

配置了远程“hello”的设备，当加入网络或者自身的元数据改变时，除了组播“hello”消息外，还发送一个远程“hello”消息到它的“本地 DP”。这个消息是作为一个 web 服务请求操作而发送的，通过使用 HTTP 绑定（在 ONVIF DP WSDL 中定义），从设备发到 DP。远程 hello 应该在“hello”消息中包含它的范围清单。

一旦“本地 DP”收到来自其他任何设备的“hello”消息时，它将立马回一个 hello 响应消息用以确认设备注册情况。

类似的，一旦设备准备离开这个网络，它应该给远程 DP 发送一个 BYE 请求。远程 DP（发现代理）通过一个 BYE 应答来确认收到了 BYE 请求。

Hello 请求和 hello 应答，BYE 请求和 BYE 应答是由一个 dp 服务实现的，它的 WSDL 描写

在 [ONVIF DP WSDL]中。

使用这些扩展，发现信息能够到达一个理想的终端。

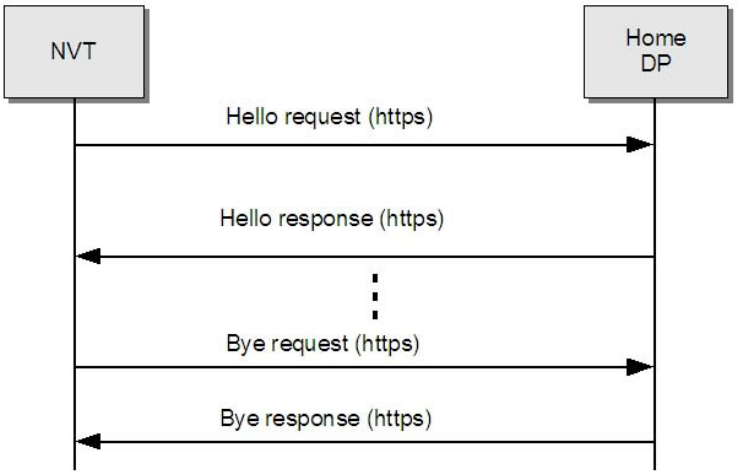


图 12：远程发现消息交换模式之间的装置（NVT 和一个家庭 DP）

7.4.4 客户端行为

对于远程发现情况，客户端需要发送探测信息到“本地 dp”，然后，客户端需要配置，使得它能够直接连接到“本地 DP”。

7.4.4.1 NVC 本地 DP 配置

配置客户端通过本地 DP 能直接探测新设备，这种情况下，本地 DP 发现服务地址应预先配置到客户端，具体的配置方法则在本规范之外。

配置了远程发现的客户端直接发送探测请求到它的“本地 DP”,这个消息是作为一个 web 服务请求操作而发送的，通过使用 HTTP 绑定（在 ONVIF DP WSDL 中定义），从客户端发到 DP。

一旦“本地 DP“从任何一个客户端哪里收到一个探测消息，它将以相应的探测匹配消息来回应，并使用正常的 WS-Discovery 的消息交互模式，看图 13 的序列图

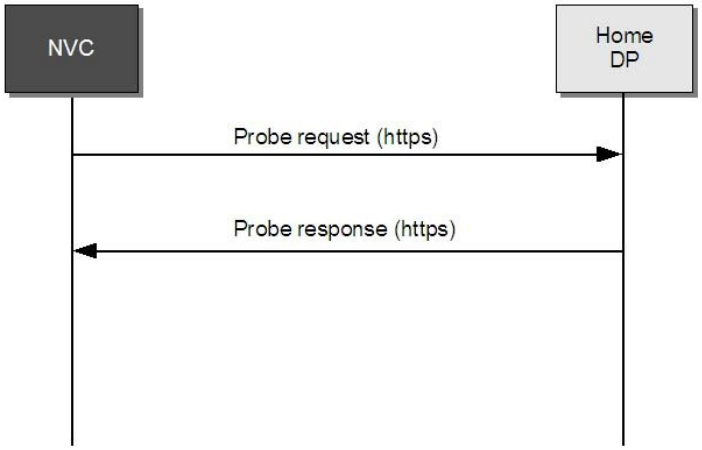


图 13：为客户(NVC)的消息序列预先配置的家庭 DP 地址

7.4.5 安全

7.4.5.1 本地发现

安全和发现被视为一对矛盾的目标。虽然发现协议背后的主要想法是公布服务存在，但还是很难排除其它端点接入该服务公告系统。WS-Discovery 不提供任何额外的接入服务（如何使用了本规范定义的其它安全机制），即使在同一局域网中。它仅仅宣布它们的存在。此外，本地发现仅仅在组播达到的范围内才起作用。因此，主要 WS-Discovery 安全的隐患是设备受到拒绝服务攻击的风险，或是隐私问题，如果在网络上隐藏设备的存在是重要的。这两个问题的风险将很大程度上取决于设备的部署环境。为了减少这些威胁，此规范引进了两种不同的发现模式，看 7.2 章节，这就是允许客户端关闭设备发现的功能，在不可发现模式下，一个设备不会用 hello 消息来宣称它的存在，也不会回应探测或处理的请求。

7.4.5.2 远程发现

在远程网络情景中，DP 部署在互联网中的，并且它是很脆弱的。应该采取额外的安全措施，来防止 DP 被攻击。远程 hello 和探测及探测匹配信息应该通过 HTTPS 发送。这种传输虽然不能防止拒绝服务的攻击，但是如果使用客户端认证，它可以防止非法设备的注册。如果拒绝服务时受到保护是主要目的话，还需要采纳其他的措施，（目前规范之外的一些措施）。

一个基于发现代理的设备，在注册设备之前应该验证设备，以确保是一个存在的合法的设备，例如通过使用客户端认证。客户端证书规定范围在现行规范范围之外。

客户端应该通过 HTTPS 发送远程探测和探测匹配的消息到 DP。DP 在回应一个探测请求之前应该验证 NVC，这可以使用 TLS 客户认证或者任何其他合适的客户身份验证机制来完成。

8 设备管理

设备服务可以分为五种不同类型的命令集：功能，网络，系统，输入输出（I/O）以及安全。这组命令可以用来获取设备的功能和配置信息，或者配置设备。设备应支持在[ONVIF DM WSDL]中规定的设备管理服务。一套基本的操作必须支持设备管理服务，其它的操作建议或者可选择性的支持该服务。在后面的命令说明中将会列出详细的规定。

8.1 功能

8.1.1获取WSDL的URL

终端可能请求到一个 URL，利用这个 URL 来检索设备完整的概要信息和 WSDL 定义。该命令返回一个 URL 链接地址；在这地址里，可以检索到必要的产品信息；这些信息都是与 WSDL 和概要定义相关的。通过使用 GetWsdlUrl 命令，设备必须能返回一个用于下载 WSDL 和概要信息的 URL。

表 9：Get WSDL URL 命令

GetWsdlUrl	请求与应答
信息名称	相应功能及参数描述
GetWsdlUrlRequest	请求信息中不含有任何信息
GetWsdlUrlResponse	应答信息包含请求的 URL xs:anyURI WsdlUrl [1][1]
错误代码	原因分析
	没有与命令相关的错误代码

8.1.2交换的功能

通过功能交换的请求与应答操作，任何终端都能够查询设备的功能。通过调用 GetCapabilities 命令，设备在应答信息中应表明设备在 ONVIF 协议所包含的所有功能。

返回的功能清单包含相关的服务地址，该服务实现了该类别中接口的操作。

表 11 描述如何理解指定的功能，除了地址的功能，还有在本规范的可选功。

表 10 ： Get Capabilities 命令

GetCapabilities	请求与应答
信息名称	信息内容以及功能

GetCapabilitiesRequest	请求信息包含对设备功能查询的请求。客户端既可以查询设备拥有的所有功能或者仅仅其中具体一种服务的功能。如果没有指定要查询的类别，那设备就会在应答信息中包含设备所有功能。 tt:CapabilityCategory Category [0][unbounded
GetCapabilitiesResponse	在应答的信息中包含一个采用 XML 分层能力结构体，用这个结构体来表示要查询设备功能 tt:Capabilities Capabilities [1][1]
错误代码	原因分析
env:Receiver ter:ActionNotSupported ter:NoSuchService	设备不支持请求信息提到的 WSDL 服务类别

表 11：GetCapabilitie 命令中的功能

类别	功能	功能描述
分析	XAddr	分析服务地址。如果此字段是空的，设备支持分析，但不支持这规则或模块接口。
	RuleSupport	表明设备支持第 17.2 节中提到接口规则和语法规则。
	AnalyticsModuleSupport	设备支持在 17.3 节中提到的场景分析模块接口节规则。
设备	XAddr	设备的服务地址
设备网络	IPFilter	表明设备支持使用 IP 过滤控制命令，（ 在 8.2.17，8.2.18，8.2.19 和 8.2.20 提到的）。
	ZeroConfiguratio	表明设备支持在 8.2.15 和 8.2.16 节定义的零配置命令
	IPVersion6	表明设备支持 IP 协议第 6 版本
	DynDNS	表明设备支持动态 DNS 配置（8.2.7 和 8.2.8 ）。

	Dot11Configuration	表明设备支持对 IEEE802.11 的配置（8.2.21 节）
设备与系统	DiscoveryResolve	表明设备能对解决请求进行响应（如 7.3.5 节所述）
	DiscoveryBye	表明设备能够发送字节信息(第 7.3.6 节所述）
	RemoteDiscovery	表明设备支持远程发现（如 7.4 节中指定）
	SupportedVersions	设备所能支持 ONVIF 标准规范的版本清单
	SystemBackup	表明设备支持系统备份和恢复（如 8.3.3 和 8.3.5 节）。
	FirmwareUpgrade	表明设备支持固件升级（第 8.3.10 节）。
	SystemLogging	表明设备支持系统日志检索 (第 8.3.11 节)。
	HttpSystemBackup	表明设备支持使用 HTTP GET 和 POST 机制进行系统备份和恢复
	HttpFirmwareUpgrade	表明设备支持使用 HTTP POST 机制进行硬件升级
	HTTPSystemLogging	表明设备支持使用 HTTP Get 进行系统日志检索(参见 8.3.2 节）
	HTTPSupportInformation	表明设备支持使用 HTTP GET 机制检索支持的信息 （参见 8.3.2）。
设备被 I/O	InputConnectors	输入连接的数。...
	RelayOutputs	继电器输出的数

	Auxiliary	表明支持清单里的辅助命令
设备安全	TLS1.0	支持传输层安全协议 TLS1.0
	TLS1.1	支持传输层安全 TLS1.1
	TLS1.2	支持传输层安全 TLS1.2
	OnboardKeyGeneration	表明设备支持板载密钥生成和创建自签名证书（参见第 8.4.8 节）。
	AccessPolicyConfig	表明设备支持检索和加载设备访问控制策略（参见第 8.4.1 和第 8.4.2）。
	X.509Token	设备支持网络安全协议 X.509 令牌 [WS-X.509Token]
	SAMLToken	表明设备支持网络服务安全的 SAML 令牌协议（WS-Security WS-SAML）
	KerberosToken	表明设备支持 WS-Security 的 Kerberos 令牌协议 [WS-KerberosToken]。
	RELToken	表明设备支持网络服务安全的 REL 的令牌协议 [WS-RELToken]。
	Dot1X	表明设备支持 IEEE 802.1X 基于端口 网络身份验证协议
	SupportedEAPMethod	设备支持 EAP 方法。这些数字与 IANA（互联网数字分配）机构注册的数字对应。
	RemoteUserHandling	表明设备是否支持远程用户处理和应答的方法（参考 8.4.21 和 8.4.22）
事件	XAddr	事件服务地址
	WSSubscriptionPolicySupport	表明设备是否支持 WS 付费政策（参考 15.1.2 节）

	WSPullPointSupport	表明设备是否支持 WS 拉点（参考第 15.1.2）
	WSPausableSubscription-ManagerInterfaceSupport	表明设备支持 WS 可暂停认购管理接口（参考 15.1.2 节）
图像	XAddr	图像服务地址
媒体	XAddr	媒体服务地址
媒体流	RTPMulticast	表明支持 UDP 的组播述（参考第 12.1.1.1)
	RTP_TCP	表明设备支持基于 TCP 的 RTP 协议，（请参阅第 12.1.1.2）。
	RTP_RTSP_TCP	指示如果设备支持 RTP / RTSP/ TCP 传输，（参考第 12.1.1.3）。
媒体类别	MaximumNumberOfProfiles	设备所支持最大媒体访问量
云台	XAddr	云台服务的地址
接收终端	XAddr	服务接收地址。
	RTP_Multicast	表明设备能够收到的 RTP 协议组播
	RTP_TCP	表明设备能够收到基于 TCP 的 RTP 协议的传输。
	RTP_RTSP_TCP	表明设备支持接收基于 RTSP/RTP /TCP 协议传输的信息
	SupportedReceivers	设备所能支持最大接收数
	MaximumRTSPURILength	实时传输流中 URL 最大允许长度

记录	XAddr	记录控制服务地址
	DynamicRecordings	表明设备支持动态创建和删除记录（参考 19.4 与 19.5 节）
	DynamicTracks	表明设备支持动态创建和删除追踪（参考见 19.9 节 19.10）。
	DeleteData	表明设备支持明确的数据删除，参阅（Fehler！ verweisquelle konntegefundenwerden..）
搜索	XAddr	记录搜索服务的地址
	MetadataSearch	表明设备支持通用搜索的记录的原数据，（参见 20.13 和 20.14 节）。
回放	XAddr	重播服务地址。
分析设备	XAddr	分析设备服务地址
显示	XAddr	显示服务的地址
显示层	FixedLayout	表明该器件具有一组特定的预定义布局
设备 I/O	XAddr	设备 I/O 服务的地址
	VideoSources	视频输入数
	VideoOutputs	视频输出数
	AudioSources	音频输入数
	AudioOutputs	音频输出数
	RelayOutputs	继电器输出数

8.2 网络

8.2.1获取主机

终端可以通过这个操作来获取设备的主机名。终端调用 GetHostname 命令，设备必须应答其主机名的配置信息。

表 12：GetHostname 指令

GetHostname	请求与应答
信息名称	相应功能以及参数描述
GetHostnameRequest	请求信息不带任何参数
GetHostnameResponse	应答信息包括： “FromDHCP”：如果主机名是通过 DHCP(动态主机分配协议)获得 “NAME”：主机名。在 DHCP 的主机名的情况下,主机名是通过从 DHCP 服务获得 xs:boolean FromDHCP [1][1] xs:token Name [0][1]
错误代码	原因分析
	没有与此命令相关错误代码

8.2.2设置主机名

终端可以通过这个操作来设置设备的主机名。终端通过调用 SetHostname 命令，来设置设备的主机名。注意：调用 SetDNS 命令可能导致先前的主机名被重新写入设备。

表 13： SetHostname 指令

SetHostname	请求与应答
信息名称	相应功能及参数描述
SetHostnameRequest	请求信息包含： “Name”：将要设置的主机名称 xs：token Name [1][1]
SetHostnameResponse	应答信息中不包含任何内容
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:InvalidHostname	设备不能接受请求信息中设定的主机名

8.2.3 获取 DNS配置

通过此操作,终端能够获取设备的 DNS 配置。终端通过调用 GetDNS 命令，设备在应答信息中返回其 DNS 配置给终端。

表 14：GetDNS 命令

GetDNS	请求与应答
信息名称	相应功能以及参数描述
GetDNSRequest	请求信息中不包含任何参数
GetDNSResponse	应答信息包含： 1.“FromDHCP”：如果是通过 DHCP 获得的 DNS 服务器。 2.“SearchDomain”：如果主机名没资格，进行域搜索. 3.“DNSFromDHCP”：如果能够通过“FromDHCP 来获取 DNS，”DNS”服务清单也通过“FromDHCP”来获取：这意味着：在 DNSFromDHCP 领域的解析地址是来源于 DHCP 以及配置状态描述 4．“DNSManual”手动给定的 DNS 服务器列表 xs:boolean FromDHCP [1][1] xs:token SearchDomain [0][unbounded] tt:IPAddress DNSFromDHCP [0][unbounded] tt:IPAddress DNSManual [0][unbounded]
错误代码	原因分析
	没有与此命令相关的错误代码

8.2.4设置DNS

通过调用 SetDNS 命令，终端可以对设备的 DNS 进行配置。

表 15 SetDNS 命令

SetDNS	请求与应答
信息名称	相应功能以及参数描述
SetDNSRequest	请求信息包括： 1.“FromDHCP”：如果是通过 DHCP 能够获得的 DNS 服务 2.“SearchDomain”：如果 hostname 是不完整，进行域搜索。3.“DNSManual”：手动 DNS 服务器列表 xs:boolean FromDHCP [1][1] xs:token SearchDomain [0][unbounded] tt:IPAddress DNSManual [0][unbounded]
SetDNSResponse	应答信息不包含任何信息
错误编码	原因分析
env:Sender ter:InvalidArgVal ter:InvalidIPv6Address	建议的 IPv6 地址是无效的。
env:Sender ter:InvalidArgVal ter:InvalidIPv4Address	建议的 IPv4 地址是无效的

8.2.5获取NTP配置信息

通过此操作，终端可以获取设备 NTP 配置。如果设备支持 NTP 的，那么终端可以通过调用 GetNTP 命令来获取设备的 NTP 服务配置。

表 16：GetNTP 指令

GetNTP	请求与应答
信息名称	相应功能以及参数描述
GetNTPRequest	请求信息不含有任何参数
GetNTPResponse	应答信息包括： 1.“FromDHCP”：如果能通过 DHCP 获得 NTP 服务。 2.“NTPFromDHCP”：如果通过“FromDHCP 来获取 NTP，”NTP”服务清单也通过“FromDHCP”来获取；这意味着：在 NTPFromDHCP 领域的地址解决是来源于 DHCP 以及配置状态描述 3.“NTPManual”：手动给 NTP 服务器列表。 xs:boolean FromDHCP [1][1] tt:NetworkHost NTPFromDHCP[0][unbounded] tt:NetworkHost NTPManual [0][unbounded]
可能的错误	原因分析
	没有与此命令相关的错误

8.2.6 对设备设置NTP

终端通过此操作来对设备的 NTP 进行配置。如果设备支持的 NTP 的，那么终端可以通过 SETNTP 命令，来对设备的 NTP 服务进行配置。

表 17 SetNTP 指令

SetNTP	请求与应答
信息名称	相应功能以及参数描述
SetNTPRequest	请求信息包含： 1.“FromDHCP”：如果通过 DHCP 获得 NTP 服务。 2.“NTPManual”：当没能通过 DHCP 获取 NTP 服务，手动创建的 NTP 服务器列表。 xs:boolean FromDHCP [1][1] tt:NetworkHost NTPManual [0][unbounded
SetNTPResponse	应答信息是一条空信息，
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:InvalidIPv4Address	建议的 IPv4 地址是无效的

env:Sender ter:InvalidArgVal ter:InvalidIPv6Address	建议的 IPv4 地址是无效的
env:Sender ter:InvalidArgVal ter:InvalidDnsName	建议的 NTP 服务名是无效的

8.2.7获取动态的DNS设置

终端调用这个操作来从设备处获取设备动态 DNS 配置。如果设备支持在 [RFC 2136] 和[RFC 4702]中指定的动态 DNS 配置，那么终端可以通过 GetDynamicDNS 命令来获得设备动态 DNS 的类型，名称以及 TTL。

表 18：GetDynamicDNS 指令

GetDynamicDNS	请求与响应
信息名称	相应功能以及参数描述
GetDynamicDNSRequest	请求信息不包含任何参数
GetDynamicDNSResponse	应答信息包含： 1.“Type”：更新类型。可能有三种类型：1 设备的本身没有更新（NOUPDATE）；2. DHCP 服务器更新（ServerUpdates）；3.设备更新（ClientUpdates）。 2.“Name”:：在设备未有更新的情况下，DNS 名称。3. “TTL”：生存时间。 tt:DynamicDNSType Type [1][1] tt:DNSName Name [0][1] xs:duration TTL [0][1]
错误代码	原因分析
	没有与这类命令相关的错误代码

8.2.8设置设备动态DNS

终端通过调用这操作来配置设备的动态 DNS。如果设备支持在[RFC2136]和[RFC4702]中的动态 DNS 配置，那么终端可以通过 SetDynamicDNS 命令来设置设备动态的 DNS 的类型，名称以及生存时间。

表 19：SetDynamicDNS 指令

SetDynamicDNS	请求与应答
信息名称	相应功能以及参数描述
SetDynamicDNSRequest	请求信息包含： 1.“Type”：更新类型。有三种类型：1 设备的期望没有更新（NOUPDATE）2 设备需要 DHCP 服务器更新（ServerUpdates）3 设备更新本

	（ClientUpdates）。 2.“Name”：在设备更新情况下，DNS 名称。 3.“TTL”：生存时间。 tt:DynamicDNSType Type [1][1] tt:DNSName Name [0][1] xs:duration TTL [0][1]
SetDynamicDNSResponse	应答信息不包含任何信息
错误代码	原因分析
	没有与这类命令相关的错误代码

8.2.9 获取网络接口配置

终端通过这个操作来获取设备的网络接口配置。终端调用 GetNetworkInterfaces 命令，设备应返回给终端其网络接口配置，这网络接口配置是已经在网络接口类型中定义好的。

表 20：GetNetworkInterfaces 指令

GetNetworkInterfaces	请求与应答
信息名称	相应功能以及参数描述
GetNetworkInterfacesRequest	请求信息是一条空信息
GetNetworkInterfacesResponse	应答信号包含一个设备的网络接口的矩阵 tt:NetworkInterfaceNetworkInterfaces[0][unbounded]
错误代码	原因分析
	没有与此命令相关错误代码

8.2.10 设置网络接口配置

通过这操作，终端可以对设备的网络接口进行配置。终端通过调用 SetNetworkInterfaces 命令来设置设备的网络接口。

为了与未知 IEEE802.11 扩展的客户端互通，如果请求信息中的 IEEE802.11 配置元素不存在，那么设备应该保留它的原来的 IEEE802.11 配置。

表 21：SetNetworkInterfaces 命令

SetNetworkInterfaces	请求与应答
信息名称	相应功能以及参数描述
SetNetworkInterfacesRequest	请求信息包括：“InterfaceToken”：要打开的网络令牌接口。“NetworkInterface”：需要配置网络接口。 tt:ReferenceToken InterfaceToken [1][1] tt:NetworkInterfaceSetConfigurationNetwork

	Interface[1][1]
SetNetworkInterfacesResponse	应答信息包括： “RebootNeeded”：表明：当网络的配置改变，需要进行从新启动 xs:boolean RebootNeeded [1][1]
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:InvalidNetworkInterface	提供的网络接口令牌不存在
env:Sender ter:InvalidArgVal ter:InvalidMtuValue	MTU 值是无效的
env:Sender ter:InvalidArgVal ter:InvalidInterfaceSpeed	不支持建议的速度
env:Sender ter:InvalidArgVal ter:InvalidInterfaceT	不支持建议的网络接口类型
env:Sender ter:InvalidArgVal ter:InvalidIPv4Address	建议的 IPV4 地址是无效的
env:Sender ter:InvalidArgVal ter:InvalidIPv6Address	建议的 IPV6 地址是无效的
env:Receiver ter:ActionNotSupported ter:InvalidDot11	不支持 IEEE 802.11 配置
env:Sender ter:InvalidArgVal ter:InvalidSecurityMode	不支持选用的安全模式
env:Sender ter:InvalidArgVal ter:InvalidStationMode	不支持选择的站模式
env:Sender ter:InvalidArgVal ter:MissingDot11	在安全配置中 IEEE 802.11 的值丢失或者不存在
env:Sender ter:InvalidArgVal ter:MissingPSK	在安全配置中 PSK 值丢失

env:Sender ter:InvalidArgVal ter:MissingDot1X	在安全配置中 IEEE 802.1X 值丢失或者根本就不存在
env:Sender ter:InvalidArgVal ter:IncompatibleDot1X	在网络接口安全配置中 IEEE 802.1X 值与网络接口不兼容

8.2.11 获取网络协议

通过操作，终端可以获取已定义在设备上网络协议，设备应该支持通过 GetNetworkProtocols 命令来返回其所配置的网络协议。

表 22：GetNetworkProtocols 命令

GetNetworkProtocols	请求与应答
信息名称	相应功能以及参数描述
GetNetworkProtocolsRequest	请求信息是一条空信息
GetNetworkProtocolsResponse	应答消息返回一个设备支持协议的矩阵。有三个已定义的协议，HTTP，HTTPS 和 RTSP。对于每个协议，可以检索以下参数： Port Enable/disable tt:NetworkProtocolNetworkProtocols[0][unbounded]
错误代码	原因分析
	没有与这类命令相关的错误

8.2.12 设置网络协议

通过操作能够对设备上定义的网络协议进行配置。设备应该支持通过 SetNetworkProtocols 命令来对已定义的网络协议的进行配置。

表 23：SetNetworkProtocols 命令

SetNetworkProtocols	请求与应答
信息名称	相应功能及参数描述
SetNetworkProtocolsReques	请求信息能够配置一个或者多个设备支持的网络协议。就目前存在三个已定义的信息：HTTP,HTTPS,RTSP；下面分别是对每个协议需要设定的参数： Port Enable/disable tt:NetworkProtocolNetworkProtocols[1][unbounde]
SetNetworkProtocolsResponse	应答信息是一条空信息
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:ServiceNotSupported	不支持提供的网络协议

8.2.13 获取默认的网关

通过这个操作，终端能够获取设备的默认网关配置信息，终端调用

GetNetworkDefaultGateway 命令来获取设备的默认网关地址。

表 24：GetNetworkDefaultGateway 命令

GetNetworkDefaultGateway	请求与应答
信息名称	相应功能以及参数描述
GetNetworkDefaultGatewayRequest	请求信息是一条空信息
GetNetworkDefaultGatewayResponse	应答信息包含： “IPv4Address：默认网关 IPV4 地址 IPV6 Address：默认的网关 IPV6 地址 tt:IPv4Address IPv4Address [0][unbounded] tt:IPv6AddressIPv6Address[0][unbounded]
错误代码	原因分析
	没有与此相关的错误

8.2.14 设置默认网关

通过这个操作，能够对设备的默认网关进行设置，设备应该支持通过 SetNetworkDefaultGateway 命令来配置其默认网关

表 25：SetNetworkDefaultGateway 命令

SetNetworkDefaultGateway	请求与回答
信息名称	描述
SetNetworkDefaultGatewayRequest	请求信息包含： “IPv4Address”： 设置默认 IPV4 网关地址 “IPv6Address”:设置默认 IPV5 网关地址 tt:IPv4Address IPv4Address [0][unbounded] tt:IPv6Address IPv6Address [0][unbounded]
SetNetworkDefaultGatewayResponse	应答信息是一条空信息
错误的代码	描述原因
env:Sender ter:InvalidArgVal ter:InvalidGatewayAddress	所提供的网关地址是无效

8.2.15 获取0配置

通过这个操作，能够获取设备的 0 配置；如果设备支持根据 [RFC3927] 的动态 IP 配置，那么设备应该支持通过 GetZeroConfiguration 命令来返回其 IPV4 0 配置地址和状态；

表 26： GetZeroConfiguration 命令

GetZeroConfiguration	请求与应答
信息名称	相应功能以及参数描述
GetZeroConfigurationRequest	请求信息是一条空信息

GetZeroConfigurationResponse	应答信息包括： “InterfaceToken”:网络接口令牌环 “Enabled”如果 0 配置被使能或者没有 “Addresses”IPV4　0 配置地址 tt:ReferenceToken InterfaceToken [1][1] xs:boolean Enabled [1][1] tt:IPv4Addresses Address [0][unbounded]
错误代码	原因分析
	没有与此命令相关的错误代码

8.2.16 设置0配置

通过这个操作，终端可以对设备进行 0 配置；如果设备支持[RFC 3927]动态的 IP 配置，那么设备应该支持通过 SetZeroConfiguration 命令来对设备的 IPv4 0 的状态和地址进行配置。

表 27：SetZeroConfiguration 命令

GetZeroConfiguration	请求与应答
信息名称	描述
GetZeroConfigurationRequest	请求信息包含： “InterfaceToken”令牌网络接口操作 “Enabled”: 0 配置使能与否 tt:ReferenceToken InterfaceToken [1][1] xs:boolean Enabled [1][1]
GetZeroConfigurationRespon se	应答信息是一条空信息
可能存在的 错误	原因分析
env:Sender ter:InvalidArgVal ter:InvalidNetworkInterface	所提供的网络接口令牌不存在

8.2.17 获取IP地址过滤

此操作可以获取设备 IP 地址过滤的配置信息。如果设备支持基于 IP 过滤规则设备访问控制（拒绝或接受的变换 IP 地址），那么设备应该支持通过 GetIPAddressFilter 命令来获取其 IPaddressfilter 配置信息。

表 28：GetIPAddressFilter 指令

GetIPAddressFilter	请求与应答
信息名称	相应功能以及参数描述

GetIPAddressFilterRequest	请求信息是一条空信息
GetIPAddressFilterResponse	应答信息包括： “Type”:设置是否允许访问过滤. “IPv4Address”: IPv4 滤波器地址 “IPv6Address”: IPv6 滤波器地址 tt:IPAddressFilterType Type [1][1] tt:PrefixedIPv4AddressIPv4Address[0][unbounded] tt:PrefixedIPv6Address IPv6Address [0][unbounded
错误代码	原因分析
	没有与此命令相关的错误代码

8.2.18 对IP地址过滤进行配置

通过这个操作，终端能够对设备的 IP 地址过滤进行配置。如果设备支持基于 IP 过滤规则的设备访问控制，那么终端就可以通过 SetIPAddress 命令来对设备 IP 过滤进行配置。

表 29：SetIPAddressFilter 指令

SetIPAddressFilter	请求与应答
信息名称	相应功能以及参数描述
SetIPAddressFilterRequest	请求信息包括： “Type”: 设置类型如果滤波器允许访问. “IPv4Address”: IPv4 滤波器地址 “IPv6Address”: IPv6 滤波器地址 tt:IPAddressFilterType Type [1][1] tt:PrefixedIPv4Address IPv4Address[0][unbounded] tt:PrefixedIPv6AddressIPv6Address[0][unbounded
SetIPAddressFilterResponse	应答信息为空
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:InvalidIPv6Address	建议的 IPV6 地址无效
env:Sender ter:InvalidArgVal ter:InvalidIPv4Address	建议的 IPV4 地址无效

8.2.19 增加IP地址过滤

通过这个操作可以增加设备的 IP 地址过滤。如果设备支持基于 IP 滤波规则的设备访问控制（排斥或者接受动态 IP 地址），那么设备应该支持调用 AddIPAddressFilte 命令来增加设备的 IP 地址过滤。

表 30：AddIPAddressFilter 指令

AddIPAddressFilter	请求与应答
信息名称	相应功能以及参数描述

AddIPAddressFilterRequest	请求信息包含： “IPv4Address”: 增加的 IPv4 滤波地址 “IPv6Address”: 增加的 IPv46 滤波地址 tt:PrefixedIPv4Address IPv4Address[0][unbounded] tt:PrefixedIPv6AddressIPv6Address[0][unbounded]
SetIPAddressFilterResponse	应答信息是一条空信息
可能的存在的错误	原因分析
env:Sender ter:InvalidArgVal ter:IPFilterListIsFull s	IP filter 清单已满，不能够继续添加 IP 滤波地址
env:Sender ter:InvalidArgVal ter:InvalidIPv6Address	建议的 IPV6 地址是无效的
env:Sender ter:InvalidArgVal ter:InvalidIPv4Address	建议的 IPV4 地址是无效的

8.2.20 移除IP地址过滤

通过这个操作能够删除设备的一个 IP 地址过滤。如果设备支持基于 IP 过滤规则的设备访问控制，那么终端可以调用 RemoveIPAddressFilter 指令来删除设备的 IP 地址过滤。

表 31：RemoveIPAddressFilter 命令

RemoveIPAddressFilter	请求与应答
信息名称	相应功能以及参数描述
RemoveIPAddressFilterRequest	请求信息包含： “IPv4Address”: 增加的 IPv4 滤波器地址 “IPv6Address”: 增加的 IPv46 滤波器地址 tt:PrefixedIPv4AddressIPv4Address[0][unbounded] tt:PrefixedIPv6AddressIPv6Address[0][unbounded]
RemoveIPAddressFilterResponse	应答信息不存在或者不包含任何信息
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:InvalidIPv6Address	请求信息中要移除的 IPV6 地址无效的
env:Sender ter:InvalidArgVal ter:InvalidIPv6Address	请求信息中要移除的 IPV4 地址是无效
env:Sender ter:InvalidArgVal ter:NoIPv6Address	请求信息要移除的 IPV6 地址不存在
env:Sender ter:InvalidArgVal	请求信息要移除的 IPV4 地址不存在

ter:NoIPv4Address	

8.2.21 IEEE 802.11配置

在这一节以及这一节下的各个小节所讲的内容只针对支持 IEEE 802.11 协议的设备，在这一节以及各个小节中，设备意味着是支持 IEEE 802.11 协议的设备。

设备应该支持 IEEE 802.11 的配置，以及对 802.11 接口中 NetworkInterfaces 方法进行响应并返回一个基于 802.11 接口的 IANA-IfTypes。

设备不会在对 GetNetworkInterfaces 应答信息中包含的任何链接元素，并且它将忽略 SetNetworkInterfaces 请求的所有链接元素。

对于每一个 IEEE 802.11 网络接口，至少有超过一个可替代的 IEEE 802.11 配置与之对应；这些配置都是是设备应该支持的。

在获取与设置网络配置元素中，任意一个 IEEE 802.11 配置元素都是被支持的。以下的信息是需要被处理的：

SSID

ation mode

Multiple wireless network configuration

Security configuration

以下操作被用来帮助管理无线配置：

Get IEEE802.11 capabilities

Get IEEE802.11 status

Scan available IEEE802.11 networks

8.2.21.1 SSID

设备应该支持对 SSID 的配置。

8.2.21.2 基站模式

设备应该支持基本的基站模式结构。

设备可以支持 ad-hoc 网络基站模式，而对于实际 ad-hoc 的配置（包括：频道数的手动配置）是本协议规范之外的；但是对于支持 ad-hoc 网络基站模式的设备，协议允许选择以及报告这种模式。

8.2.21.3 多种无线网络配置

每一个 IEE802.11 配置都应该用一个别名进行区分。在网络接口配置中，别名应该是唯一的。客户端在 SetNetworkInterfaces 请求中信息中应该提供别名。如果客户端想更新存在的无线配置，那么应该用与网络接口相同的别名。包含别名在内的无线网络配置应该唯一存在，因为别名也是网络配置的一部分。

在多个可选择的 IEEE802.11 配置中，设备应该能够通过选择优先值来给予考虑，往往

越大的值其优先性也就越大。如果配置中优先值丢失，那么就会被认为是拥有最低的优先级。如果几个无线配置拥有一样的优先级，那么在这些配置中的优先级顺序就没有被定义了。

设备使能 IEEE 802.11 配置的优先级列表的算法是本协议规范之外的事情了。

8.2.21.4 安全配置

安全配置包含选择安全模式以及进行与模式相关的配置。以下的安全模式被协议所允许：

None

PSK (Pre Shared Key) (WPA- and WPA2-Personal)

IEEE 802.1X-2004 (WPA- and WPA2-Enterprise)

协议允许设备选择或者报告 WEP 安全模式，但是对于 WEP 模式配置不在本协议规范之内的事情

对于数据的保密性以及其完整性，设备应当按照[IEEE 802.112007]规范支持 CCMP 算法以及设备还可以支持 TKIP 算法。

算法可以被手动或者自动选择。在手动选择模式中，相同的算法应该成对和组密码进行使用。为了能够支持其他算法，一个“Extended”值是可用的。

设备应该支持手动或者自动选择模式。

8.2.21.4.1 None 模式

设备应该支持“None”安全模式。

8.2.21.4.2 PSK 模式

设备应该支持 PSK 安全模式。

为了减小对 PSK 的危害，设备不应该传输任何 PSK 给客户端，而且不应该在对调用 GetNetworkinterfaces 操作的应答中返回 PSK.

在 PSK 安全模式中，可以运用以下规则来添加无线配置：

1.客户端在 SetNetworkInterfaces 请求中应该包含一个 PSK 值。

2.设备应该检查确保 PSK 值是被提供，如果没有提供，设备返回一个错误。

在 PSK 安全模式中，当需要对无线配置进行更新时，按以下规则进行：

1.如果客户想保留 PSK 当前的值，那么在 SetNetworkInterfaces 请求的信息中不应该包含 PSK 的值

2.如果设备接收 SetNetworkInterfaces 请求信息中不包含 PSK 的值，那么设备保留其以前用的 PSK 值。

[IEEE 802.11-2007]标准声明：应当用一些频带外的方法将 PSK 分发到 SAT 上面。在 ONVIF 协议的安全策略强调 PSK 应该受到充分的保护。

8.2.21.4.3 IEEE 802.1X-2004 模式

设备应该支持 IEEE802.1x 安全模式。对于 IEEE802.1x-2004 安全模式的详细要求请参考[IEEE 802.1X configuration

8.2.21.5 获取 DOT11 的性能

通过这个操作,设备返回一个 IEEE802.11 的功能能，参见表 33.设备支持这个操作

表 32：GetDot11Capabilities

GetIEEE802.11Capabilities	请求与应答
信息名称	相应功能以及参数描述
GetDot11CapabilitiesRequest	请求信息是空信息
GetDot11Capabilites-Response	tt:Dot11Capabilities Capabilities [1][1]
可能的存在的错误	原因分析
env:Receiver ter:ActionNotSupported ter:InvalidDot11	不支持 IEEE802.11 配置

表 33：IEEE802.11 capabilities

Capability	具体描述
TKIP	表明设备支持 TKIP 算法
ScanAvailableNetworks	表明设备支持 ScanAvailableIEEE802.11Networks 操作
MultipleConfiguration	表明设备多替代性性 IEEE 802.11 配置
AdHocStationMode	表明设备支持 ad-hoc 基站模式
WEP	表明设备支持 WEP 安全模式

8.2.21.6 GetIEEE802.11 状态

通过这个操作能够返回一个无线网络接口状态 信息，这设备应该支持这个命令，以下是可能返回的状态信息：

SSID (shall)

BSSID (should)

Pair cipher (should)

Group cipher (should)

Signal strength (should)

Alias of active wireless configuration (shall)

表 34：GetDot11Status

GetDot11Status	请求与应答
信息名称	相应功能以及参数描述
GetDot11StatusRequest	tt:ReferenceToken InterfaceToken [1][1]
GetDot11StatusResponse	tt:Dot11Status Status [1][1]

可能的存在的错误	原因分析
env:Receiver ter:ActionNotSupported ter:InvalidDot11	IEEE802.11 配置不支持
env:Sender ter:InvalidArgVal ter:NotDot11	接口不是 IEEE802.11 接口
env:Sender ter:InvalidArgVal ter:InvalidNetworkInterface	提供的网络接口令牌不存在
env:Receiver ter:Action ter:NotConnectedDot11	IEEE 802.11 网络没有连接上

8.2.21.7 扫描可用的 IEEE802.11 网络

通过这个操作返回一个设备周围无线网络的清单。设备应该支持这种操作。以下是每个网络可能返回的状态：

SSID (shall)

BSSID (should)

Authentication and key management suite(s) (should)

Pair cipher(s) (should)

Group cipher(s) (should)

Signal strength (should)

表 35：ScanAvailable802.11Networks

ScanAvailable802.11Networks	请求与应答
信息名称	相应功能以及参数描述
ScanAvailableDot11NetworksRequest	tt:ReferenceToken InterfaceToken [1][1]
ScanAvailableDot11NetworksResponse	tt:Dot11AvailableNetworksNetworks[0][unbounded
错误代码	原因分析
env:Receiver ter:ActionNotSupported ter:InvalidDot11	不支持 IEEE802.11 的配置
env:Sender ter:InvalidArgVal ter:NotDot11	接口不是 IEEE802.11 接口
env:Sender ter:InvalidArgVal ter:InvalidNetworkInterface	提供的网络接口令牌不存在
env;Receiver ter:ActionNotSupported ter:NotScanAvailable	设备不支持 ScanAvailableDot11Networks

8.3 系统

8.3.1设备信息

终端通过这个操作来获取设备信息，如：设备制造商，模型，以及固件版本；终端通过调用 GetDeviceInformation 命令，设备应该应答其设备信息。

表 36：GetDeviceInformation 指令

GetDeviceInformation	请求与应答
信息名称	相应功能以及参数描述
GetDeviceInformationRequest	请求信息是一条空信息空信息
GetDeviceInformationResponse	应答信息的包含以下： xs:string Manufacturer [1][1] xs:string Model [1][1] xs:string FirmwareVersion [1][1] xs:string SerialNumber [1][1] xs:string HardwareId [1][1]
错误代码	原因分析
	没有与此命令相关的错误代码

8.3.2获取系统的URL

这个操作可用于检索 URLS;这个 URIS 可以通过 HTTP 方式下载系统信息。通过 URL 可以下载以下的系统信息：

1.系统日志：很多系统日志会以不同格式返回。至于返回系统日志确切的格式不是这个协议所规范的。

2.支持信息：从设备处获得任意设备的诊断信息；至于返回诊断信息的格式不是本协议所能规范的

3.系统备份：接收到的文件是一个备份文件，该备份文件可以用来恢复上一次设备配置信息；备份文件确切的格式不是这个协议所规范。

如果设备允许检索系统日志，支持信息或者系统备份数据，那么就可以通过 HTTP GET 使其成为可能；如果设备支持这些，那么设备就必须支持 GetSystemUris 命令进行这些操作。

表 37：GetSystemUris 命令

GetSystemUris	请求与应答
信息名称	相应功能以及参数描述
GetDeviceInformationReques	请求信息是一条空信息
GetSystemUrisResponse	应答信息包含了：一个能用来下载不同信息成分的的 URL： tt:SystemLogUriList SystemLogUris [0][1] xs:anyURI SupportInfoUri [0][1] xs:anyURI SystemBackupUri [0][1]

错误代码	原因分析
	没有与此命令相关的错误

8.3.3备份

通过这个操作可以获得设备备份配置文件，终端调用 GetSystemBackup 命令，设备应该返回一个设备配置备份文件。返回的备份文件与一个用二进制表示的名字和互联网媒体类型相关；至于确切的格式已经不是这个协议所能规范的了。

通过 MOTM 来传输备份配置文件。

表 38：GetSystemBackup 指令

GetSystemBackup	请求与应答
信息名称	相应功能以及参数描述
GetSystemBackupRequest	请求信息为空信息
GetSystemBackupResponse	应答信息包含了：备份的配置文件 tt:BackupFile BackupFiles [1][unbounded]
错误代码	原因分析
	没有与此命令相关的错误代码

8.3.4恢复

这个操作可以通过早先从设备接收到的备份文件来恢复设备的配置信息；设备应该支持通过 RestoreSystem 命令来恢复设备的配置文件。至于备份配置文件具体的确切格式已经不是本协议所能规范的了；如果设备支持这个命令，那么设备就支持通过 GetSystemBackup 命令来接收备份文件来恢复配置信息。
通过 MOTM 来传输备份配置文件。

表 39: RestoreSystem 指令

RestoreSystem	请求与应答
信息名称	相应功能以及参数描述
RestoreSystemRequest	请求信息包含系统备份文件： tt:BackupFile BackupFiles [1][unbounded]
RestoreSystemResponse	应答信息是一条空信息
可能的存在的错误	原因分析
env:Sender ter:InvalidArgVal ter:InvalidBackupFile	备份文件无效

8.3.5开始恢复系统

这个操作能够通过 HTTP POST 机制，并从备份的配置数据中初始化并恢复设备系统。然后设备返回一个用来上传备份文件的 HTTP URL 作为对这个命令的响应；当 HTTP POST 操作一旦完成，恢复就开始了；设备应该支持通过 StartSystemRestore 命令来进行系统恢复。至于具体备份配置文件的格式已经不是这协议所能规范的了。

通过 HTTP 进行系统恢复会按照以下几步实现：

1 客户端调用 StartSystemRestore 命令

2 服务端回应一个用于上传的 URI

3.客户端通过 HTTP POST 上传配置数据到 URL

4.服务端采用上传的配置，重启（如果需要）

如果由于上传的是无效的文件导致系统恢复失败，这 HTTPPOST 将会应答一个“415 Unsupported Media Type”；如果系统恢复失败是由于设备的错误，这 HTTP POST 将会应答一个“500 Internal Server Error”。

在 HTTP POST 中，内容类型头的值的请求应该是“application/octetstream”

表 40：StartSystemRestore 命令

StartSystemRestore	请求与应答
信息名称	相应功能以及参数描述
StartSystemRestoreRequest	请求信息为一条空信息
StartSystemRestoreResponse	应答信息包含： 1.对于用于下载系统配置文件 URL 2.一个可选的持续时间，这个时间用来指示设备期望在下载后多久不可用 xs:anyURI UploadUri [1][1] xs:duration ExpectedDownTime [0][1]
错误代码	原因分析
	没有这 此命令相关的错误代码

8.3.6获取系统日期以及时间

这种操作可以用来获取设备系统时间以及系统日期。设备应该支持通过 GetSystemDateAndTime 命令来返回一个夏令设置时间或者人工系统时间或者这 NTP 时间。虽然 UTCDATETIME 信息被标记为可选项，设备还是应该提供一个 UTCDATETIME 信息，这主要是确保能够向后兼容。

表 41： GetSystemDateAndTime 命令

GetSystemDateAndTime	请求与应答
信息名称	相应功能以及参数描述
GetSystemDateAndTimeRequest	请求信息是一条空信息
GetSystemDateAndTimeResponse	应答信息包含设备的日期以及时间信息： “DateTimeType”: 如果系统的时间和日期是通过手动

	或者 NTP 设定的 “DaylightSavings”: 夏令时打开或者关闭 "TimeZone": 在 POSIX 1003.1 定义的时区（参见 8.3 节） “UTCDateTime”: 在 UTC 中的日期和时间. “LocalDateTime”: 设备当地的时间 tt:SetDateTimeType DateTimeType [1][1] xs:boolean DayLightSavings [1][1] tt:TimeZone TimeZone [0][1] tt:DateTime UTCDateTime [0][1] tt:DateTime LocalDateTime [0][1]
错误代码	原因分析
	没有这 此命令相关的错误代码

8.3.7设置系统日期以及时间

这个操作能够设置设备的系统日期和时间。设备应该支持通 SetSystemDateAndTime 命令来配置设备的夏令设置以及手动系统的日期和时间或者表示 NTP 的时间。

如果系统时间和日期是手动设置的，那么客户端在请求信息中就应该包含 UTCDateTime 和 LocalDateTime

表 42：SetSystemDateAndTime 指令

SetSystemDateAndTime	请求与应答
信息名称	相应功能以及参数描述
SetSystemDateAndTimeRequest	请求的信息包含了对于设备要设置的日期以及时间： “DateTimeType”: 如果系统时间个日期是手动设置的或者通过 NTP 设置的 “DaylightSavings”:夏令时打开或者关闭 "TimeZone": 在 POSIX 1003.1 定义的时区（参见 8.3 节） “UTCDateTime”: 在 UTC 中的时间和日期. 如果日期时间类型是 NTP，UTCDateTime 就没有意义. tt:SetDateTimeType DateTimeType [1][1] xs:boolean DayLightSavings [1][1] tt:TimeZone TimeZone [0][1] tt:DateTime UTCDateTime [0][1]
SetSystemDateAndTime-Response	应答信息是一个空信息
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:InvalidTimeZone	请求信息中设置的 time zone 域无效
env:Sender ter:InvalidArgVal ter:InvalidDateTime	请求信息中设置的 时间或者日期 无效

8.3.8 出厂默认配置

这个操作可以重载设备在出厂时的默认值。设备应该支持通过 SetSystemFactoryDefault 命令来重载设备在出厂时软件和硬件默认参数值。这就意味着软件出厂的默认参数与具体的产品以及具体的供应商有关。软件出厂默认的操作作用还没有被充分的定义。然而，必须保证设备在配置复位后，能够使用以前使用的 IP 地址访问。这就意味着基本的网络设置如：IP 地址，子网和网关或者 DHCP 这些设置是不能在软件重置中改变的。

表 43：SetSystemFactoryDefault 指令

SetSystemFactoryDefault	请求与应答
信息名称	相应功能以及参数描述
SetSystemFactoryDefaultRequest	请求的信息包含了设备在出厂时执行的默认配置. “Hard”: 所有的参数设置成产品出厂前的默认值 “Soft”: 除了设备供应商以外的参数都设置出厂前的默认配置 tt:FactoryDefaultType FactoryDefault [1][1]
SetSystemFactoryDefaultResponse	应答信息是一条空信息
错误代码	原因分析
	没有与此命令相关的错误代码

8.3.9 固件升级

这个操作可以用来升级设备固件版本。如果成功了进行了固件升级，那么会在设备重启之前给出一个应答信息。设备应该支持通过 UpgradeSystemFirmware 命令来进行固件升级。至于固件数据的具体格式已不是这个协议所能规范的。
固件通过 MTOM 进行传输。

表 44：UpgradeSystemFirmware 命令

UpgradeSystemFirmware	请求与应答
信息名称	相应功能以及参数描述
UpgradeSystemFirmware-Request	请求信息包含了被用来升级的固件。固件升级是“软件”那么就意味着所有参数就要保持为当前值 tt:AttachmentData Firmware [1][1]
UpgradeSystemFirmware-Response	应答的信息包含了一条向客户端报告的字符串信息，如：“Upgrade successful, rebooting in x seconds.” xs:string Message [1][1]
可能的存在的错误	原因分析

env:Sender ter:InvalidArgs ter:InvalidFirmware	固件无效，或者不被设备支持
env:Receiver ter:Action ter:FirmwareUpgrade-Failed	固件升级失败

8.3.10 开始固件升级

这个操作可以通过 HTTP POST 机制来初始化一个固件升级。然后通过返回一个 HTTP 的 URL 来对升级命令进行响应，这个 http url 是用来上传升级文件的；实际上在 HTTP POST 操作一旦完成就进行升级操作；设备应该支持通过 StartFirmwareUpgrade 命令进行固件升级，至于固件文件数据的格式已不是这个协议所能规范的了

可以通过如下几步进行固件的升级：

1.客户端调用 StartFirmwareUpgrade 命令。

2.服务器通过响应一个用于上传 URL 和延迟值。

3.如果服务器指定延时，那么客户端选择持续等待。

4.客户端通过 HTTPPOST 上传一个固件图像到 URL。

5. 通过上传镜像文件,服务器重编自身程序，然后从新启动。

如果因为升级文件无效使得固件升级失败，那么这 HTTP POST 将会应答“415 Unsupported Media Type”,如果因为设备错误导致固件升级失败,那么 HTTP POST 将响应一个“500 Internal Server Error”

在请求信息中 HTTP POST 中内容类型头的值的是“application/octetstream”。

表 45：StartFirmwareUpgrade 指令

StartFirmwareUpgrade	请求与应答
信息名称	相应功能以及参数描述
StartFirmwareUpgrade-Request	请求信息是一条空信息
StartFirmwareUpgrade-Response	应答的信息包含： 上传固件文件的 URL 选择性的时延；客户端应当等待一定时间在进行初始化固件上载之前 表明设备在期待在上载固件文件完成后多久后不可用 xs:anyURI UploadUri [1][1] xs:duration UploadDelay [0][1] xs:duration ExpectedDownTime [0][1]
错误代码	原因分析
	没有与此命令相关的错误

8.3.11 获取系统日志

这个操作可以从设备中获取系统日志。设备应该支持通过 GetSystemLog 命令来获取系统日志信息。至于系统日志确切的格式已经不是本协议所规范的。

系统日志信息被传送通过 MTOM 或者一个字符串。

表 46：GetSystemLog 指令

GetSystemLog	请求与应答
信息名称	相应功能以及参数描述
GetSystemLogRequest	请求信息包含了系统日志检索的类型。这支持的日志信息类型被定义成两种类型： “System”：系统日志 “Access”：客户端访问日志 tt:SystemLogType LogType [1][1]
GetSystemLogResponse	应答的信息包含请求的系统日志信息，设备可以选择以附件或者字符串返回的日志信息否用二进制信息。 tt:AttachmentData Binary [0][1] xs:string String [0][1]
可能的存在的错误	原因分析
env:Sender ter:InvalidArgs ter:AccesslogUnavailable	没有可访问的日志信息
env:Sender ter:InvalidArgs ter:SystemlogUnavailable	没有可用的访问日志信息

8.3.12 获取支持信息

这个操作可以获得任意设备的诊断信息。设备可以支持通过 GetSystemSupportInformation 命令来检索设备的诊断信息。至于检索信息确切的格式已不是本协议所规范的。

检索信息以附件或者字符串的形式通过 MTOM 传输。

表 47GetSystemSupportInformation 命令

GetSystemSupportInformation	请求与应答
信息名称	相应功能以及参数描述
GetSystemSupportInformationRequest	请求信息是空信息
GetSystemSupportInformationResponse	应答信息包含了设备的支持信息，设备能够选择将支持信息是以二进制或者字符串的形式进行应答 tt:AttachmentData BinaryFormat [0][1] xs:string StringFormat [0][1]
可能的存在的错误	原因分析

env:Sender ter:InvalidArgs ter:SupportInformationUnavailabl	没有可用的支持信息

8.3.13 重启

通过这个操作能够重启一个设备。在设备重启之前，应答信息就会被发送回来。设备支持通过 SystemReboot 命令来重启系统。

表 48：SystemReboot 指令

SystemReboot	请求与应答
信息名称	相应功能以及参数描述
SystemReboot	请求信息是空信息
SystemRebootResponse	应答信息包含了一个字符串应答信息，这个字符串信息向客户端返回报告信息；如：“Rebooting in xseconds.” txs:string Message [1][1]
错误代码	原因分析
	没有与此命令相关的错误代码

8.3.14 获取范围参数

这个操作可以获取设备的范围参数。这范围参数被用在设备发现上，用于匹配一个探测信息，参考第七章。范围参数有如下两种不同类型：

1.固定的

2.可配置的

固定的范围参数不能够通过设备管理接口进行修改，因为固定的范围参数在设备固件配置中作为一个永久性的设备特征。在应答的信息返回一个范围参数清单，清单中包含了范围类型。可以通过范围设置与配置范围参数操作来进行范围参数配置，参见 8.3.14 和 8.3.15 节。设备应该支持通过 GetScopes 命令来检索发现的范围参数。虽然一些范围参数是固定的，客户总是希望设备响应一个范围清单。

表 49: GetScopes 命令

GetScopes	请求与应答
信息名称	相应功能以及参数描述
GetScopesRequest	请求信息是空信息
GetScopesResponse	应答信息包含了一个在设备范围定义的 URL 清单，也可以参考第七章的范围定义 tt:Scope: Scopes [1][unbounded]
可能的存在的错误	原因分析
env:Receiver ter:Action ter:EmptyScope	范围清单是是空的

8.3.15设置范围参数

这个操作可以设置设备的范围参数。在设备发现中，范围参数用来匹配一个探测信息。这个操作可以替换所有可配置的范围参数。（除了固定的参数）如果不想替换所有参数，那么可以通过添加命令来进行配置。设备应该支持通过 SetScopes 命令来对设备的范围参数进行配。

表 50：SetScope 命令

SetScopes	请求与应答
信息名称	相应功能以及参数描述
SetScopesRequest	请求信息包含一个已经定义的设备范围的 URLS 清单。参见第七章 xs:anyURI: Scopes [1][unbounded]
SetScopesResponse	应答信息是一条空信息
可能的存在的错误	原因分析
env:Sender ter:OperationProhibited ter:ScopeOverwrite	设置的范围参数覆盖了固定参数，命令排斥
env:Receiver ter:Action ter:TooManyScopes	请求信息包含的范围清单超出支持的范围数

8.3.16 添加范围参数

这个操作可以对设备添加新的可配置的范围参数。在设备发现中，范围参数被用来匹配探测信息，参见第七章。设备应该支持通过 AddScopes 命令来添加发现范围参数。

表 51：AddScopes 命令

AddScopes	请求与应答
信息名称	相应功能以及参数描述
AddScopesRequest	请求信息包含一个用于添加进已存在的配置范围清单的 URL 清单；参见第七章 xs:anyURI:ScopeItem [1][unbounded]
AddScopesResponse	应答信息是空信息
可能的存在的错误	原因分析
env:Receiver ter:Action ter:TooManyScopes	请求信息设置的范围清单超出范围支持的数量

8.3.17 移除范围参数

通过这个操作可以删除设备可配置的范围参数。在设备发现中，范围参数被用在匹配一个探测信息。参见第七章，设备应该支持通过 RemoveScopes 命令删除设备发现范围参数。

表 52：RemoveScopes 命令

RemoveScopes	请求与应答
信息名称	相应功能以及参数描述
RemoveScopesRequest	请求信息包含一个 URL 清单； xs:anyURI:ScopeItem [1][unbounded]
RemoveScopesResponse	应答信息包含一个从设备范围参数移除的 URL 清单 xs:anyURI: ScopeItem [0][unbounded]
可能的存在的错误	原因分析
env:Receiver ter:Action ter:TooManyScopes	企图移除固定的范围参数，命令排斥
env:Sender ter:InvalidArgVal ter:NoScope	请求移除信息包含一个不存在的范围

8.3.18 获取发现模式

这个操作可以获取设备的发现模式。参见 7.2 节不同设备定义的发现模式。设备应该支持通过 GetDiscoveryMode 命令来检索设备的发现模式。

表 53：GetDiscoveryMode 指令

GetDiscoveryMode	请求与应答
信息名称	相应功能以及参数描述
GetDiscoveryModeRequest	请求信息是一条空信息
GetDiscoveryModeResponse	应答信息包含当前发现模式的设定 可发现和不可发现 tt:DiscoveryMode: DiscoveryMode [1][1]
错误代码	原因分析
	没有与此命令相关的错误

8.3.19 设置发现模式

通过这个操作可以设置设备的发现模型。参见 7.2 对各种不同发现模式的定义。设备应该支持通过 SetDiscoveryMode 命令来对设备的发现模式进行定义。

表 54：SetDiscoveryMode 命令

SetDiscoveryMode	请求与应答
信息名称	相应功能以及参数描述
SetDiscoveryModeRequest	请求信息包含对设备发现模式设定：显露或者非显露 tt:DiscoveryMode: DiscoveryMode [1][1]
SetDiscoveryModeResponse	应答信息是一条空信息
错误代码	原因分析
	没有与此命令相关的错误代码

8.3.20 获取远程发现方式

这个操作可以获取设备的远程发现方式。参见 7.4 节对远程发现的定义。一个支持远程发现的设备应当支持通过 GetRemoteDiscoveryMode 命令来检索设备远程发现方式。

表 55：GetRemoteDiscoveryMode 命令

GetRemoteDiscoveryMode	请求与应答
信息名称	相应功能以及参数描述
GetRemoteDiscoveryMode-Request	请求信息是空信息
GetRemoteDiscoveryMode-Response	应答信息包含当前远程发现方式的设定：可发现和不可发现 tt:DiscoveryMode: RemoteDiscoveryMode [1][1]
错误代码	原因分析
	没有与此命令相关的错误代码

8.3.21 设置远程发现方式

通过这个操作可以设置设备远程发现方式。参见 7.4 节远程发现以及远程拓展定义。支持远程发现的设备，同样应当支持通过 SetRemoteDiscoveryMode 命令来设置设备远程发现方式。

表 56：SetRemoteDiscoveryMode 指令

SetRemoteDiscoveryMode	请求与应答
信息名称	相应功能以及参数描述
SetRemoteDiscoveryMode-Request	请求信息包含远程发现方式的设定：显露或者非显露 tt:DiscoveryMode: RemoteDiscoveryMode [1][1]

SetRemoteDiscoveryMode-Response	应答信息是一条空信息
错误代码	原因分析
	没有与此命令相关的错误代码

8.3.22 获取远程DP地址

通过这个操作可以获取设备远程的 DP 地址或者设备地址。如果设备支持远程发现，就像 7.4 节所定义的那样。那么设备应该支持通过 GetDPAddresses 命令来检索远程 DP 地址。

表 57：GetDPAddresses 指令

GetDPAddresses	请求与应答
信息名称	相应功能以及参数描述
GetDPAddressesRequest	请求信息是一条空信息
GetDPAddressesResponse	应答信息包含配置的远程 DP 地址或者地址。如果没有远程的 DP 地址配置，那么将返回一个空的清单 tt:NetworkHost: DPAddress [0][unbounded]
错误代码	原因分析
	没有与此命令相关的错误代码

8.3.23 配置远程DP地址

通过这个操作可以对设备远程的 DP 地址或者地址进行配置。如果设备支持远程发现，正如 7.4 节所讲的那样；设备应该支持通过 SetDPAddresses 命令来配置设备的远程 DP 地址

表 58：SetDPAddresses 指令

SetDPAddresses	请求与应答
信息名称	相应功能以及参数描述
SetDPAddressesRequest	请求的信息包含对设备配置的 DP 地址或者地址 tt:NetworkHost: DPAddress [0][unbounded]
SetDPAddressesResponse	应答信息是一条空信息
错误代码	原因分析
	没有与此命令相关的错误代码

8.4 安全

在这一节包含了一系列的安全管理操作。这些操作对于网络攻击非常敏感；以及为了不危及设备，通过适当的授权级别来对这些操作进行保护。

8.4.1获取访问策略

访问不同的服务以及服务子集应该服从于访问控制。这 WS-security 架构给了终端认证提供保障。通过安全访问策略，从而决定授权发生。本协议标准并不强制指定任何特定的策略描述格式或安全政策，但是这些都取决于设备制造商或系统供应商选择的策略和策略描述格式。然而，通过一些命令能够请求一个安全策略。如果设备支持基于 WS-Security 认证的访问策略，那么设备就应该支持这命令

表 59: GetAccessPolicy 指令

GetAccessPolicy	请求与应答
信息名称	相应功能以及参数描述
GetAccessPolicyRequest	请求信息是空信息
GetAccessPolicyResponse	应答信息包含请求的策略文件 tt:BinaryData PolicyFile [1][1]
错误代码	原因分析
env:Receiver ter:Action ter:EmptyPolicy	设备的访问策略文件不存在或者是空的

8.4.2 设置访问策略

通过这个命令来设置设备的安全访问策略（关于安全访问的策略的详细信息请参考 8.4.1 节的 Get comd 命令），如果设备支持基于 WS-Security 认证的安全访问策略，那么设备应该支持这命令。

表 60：SetAccessPolicy 指令

SetAccessPolicy	请求与应答
信息名称	相应功能以及参数描述
SetAccessPolicyRequest	请求信息包含设置的策略文件 tt:BinaryData PolicyFile [1][1]
SetAccessPolicyResponse	应答信息是空信息
可能的存在的错误	原因分析
env:Sender ter:InvalidArgs ter:PolicyFormat	由于不知道策略文件格式，请求信息的策略不能够被设置

8.4.3 获取用户

通过这个操作，可以列举所有已注册用户以及用户对设备操作的相应的操作等级。设备应该支持通过 GetUsers 命令来检索已注册设备用户以及作为用户权限。

表 61：GetUsers 指令

GetUsers	请求与应答
信息名称	相应功能以及参数描述
GetUsersRequest	请求信息是一条空信息]
GetUsersResponse	应答信息包含了一个用户清单以及作为用户的相应凭据 每一行包括： 用户名称（用户的密码不会包含在相应中） 用户等级 tt:User: User [0][unbounded]
错误代码	原因分析
	没有与此命令相应的错误代码

8.4.4 创建用户

这个操作可以创建一个新的设备用户以及新用户相应权限，参见 5.12 用户权限定义。设备应该支持通过 CreateUsers 命令来创建设备用户以及这些用户相应的操作权限。要么是所有的用户被成功的创建或者返回一个错误信息，没有创建任何用户。

建议遵守 ONVIF 协议的设备支持 28 字节的密码长度，因为客户可能遵循密码的求导机制，而这种机制将导致 passwordequivalent' of length 28 bytes

表 62：CreateUsers 指令

CreateUsers	请求与应答
信息名称	相应功能以及参数描述
CreateUsersRequest	请求信息包含对创建一个新用户的所需参数元素 每条目录包含： 用户名 密码 用户等级 tt:User: User [1][unbounded]
CreateUsersResponse	应答信息是一条空信息
可能的存在的错误	原因分析
env:Sender ter:OperationProhibited ter:UsernameClash	创建的用户名已经存在
env:Sender ter:OperationProhibited ter:PasswordTooLong	设置密码太长
env:Sender ter:OperationProhibited ter:UsernameTooLong	用户名太长

env:Sender ter:OperationProhibited ter:Password	密码太弱
env:Receiver ter:Action ter:TooManyUsers	超出了最大支持的用户数量
env:Sender ter:OperationProhibited ter:AnonymousNotAllowed	未设置用户等级
env:Sender ter:OperationProhibited ter:UsernameTooShort	用户设置的用户名太短

8.4.5 删除用户

这个操作可以删除设备上用户以及他们的操作权限，参见 5.12 节对用户权限的定义。设备应当支持通过 DeleteUsers 命令来删除设备用户以及这些用户权限。设备可以有一个或者更多不能删除的固定的用户。对于这个命令，要么所有用户都成功的被删除要么返回一个错误信息，没有任何用户被删除。

表 63：DeleteUsers 指令

DeleteUsers	请求与应答
信息名称	相应功能以及参数描述
DeleteUsersRequest	请求信息包含要删除的用户名称 xs:string: Username [1][unbounded]
DeleteUsersResponse	应答信息是空信息
可能的存在的错误	原因分析
env:Sender ter:InvalidArgVal ter:FixedUser	用户名不被识别
env:Sender ter:InvalidArgVal ter:FixedUser	删除的用户是固定用户

8.4.6 对用户进行配置

通过这个操作可以更新设备上一个或者多个用户的权限以及类别。设备应该支持通过 SetUser 命令来更新设备用户以及他们的权限。对于这条命令，要么所有修改请求被成功处理，要么返回一条错误信息，没有更改请求被处理。

表 64：SetUser 指令

SetUser	请求与应答
信息名称	相应功能以及参数描述
SetUserRequest	请求信息包含一个要更新用户清单以及用户权限凭据 用户名称

	密码 用户等级 tt:User: User [1][unbounded]
SetUserResponse	应答信息是空信息
可能的存在的错误	原因分析
env:Sender ter:InvalidArgVal ter:UsernameMissing	用户名不被识别
env:Sender ter:OperationProhibited ter:PasswordTooLong	设置密码太长
env:Sender ter:OperationProhibited ter:PasswordTooWeak	密码太弱
env:Sender ter:OperationProhibited ter:AnonymousNotAllowed	用户等级不允许

8.4.7 IEEE 802.1X配置

这个协议定义以下参数将作为 IEEE 802.1X 配置参数：

- 令牌配置

这个参数与 IEEE 802.1X 配置参数相关，它被定义在'Dot1XConfigurationToken' in [ONVIF Schema]；对于在编写源代码的时候，Dot1X 命名的习惯被用来更好的读取 schema 元素

- EAP 识别

这个参数表明申请链接 IEEE 802.1X 的用户名，这个在【ONVIF Schema】定义为“identity”。

- EPA 方式

这个参数表明被使用的认证方式，这个参数在[ONVIF Schema]被定义为 EAPMethod'。

- CA 证书 ID

这个参数表示用于核实认证服务 CA 执照的 ID，这个在[ONVIF Schema].被定义为 CACertificateID。

各自选定的 EAP 方法的配置参数

根据选择 EAP 方法，一些如下相关参数是被需要的：

[EAP-MD5], [EAP-PEAP/MSCHAP-V2], [EAP-TTLS types]：认证密码，以便认证服务能使用指定的密码验证设备用户（设备）。[EAP-MD5]不适用于 802.11 (WPA-Enterprise)规范。

[EAP-TLS]:客户端的证书编号，RADIUS 服务能够通过指定的证书来确认用户

在进行某网络接口配置的安全配置时，IEEE 802.1X 参数将会被用到。对于细节上的东西，请参考 8.2.10。

这个规范假定在 IEEE 802.1X 管理网络之外，完成在设备的 IEEE 802.1X 配置。为了对 IEEE 802.1X 进行重新设置，这里同样的假设设置在 802.1X 管理网络之外进行。

注意：在 ONVIF2.0 协议对 IEEE 802.1X 支持只限制在 IEEE 802.11 接口中。

8.4.7.1 创建 IEEE802.1X 配置

通过这个操作可以创建对设备 802.1X 的配置参数。如果设备支持 802.1x，那么设备就应该支持这个命令。如果设备收到一个已经存在的令牌请求，那么设备就应该响应'一个 ter:ReferenceToken '来表明有配置的冲突。

表 65：CreateDot1XConfiguration 命令

CreateDot1XConfiguration	请求与应答
信息名称	相应功能以及参数描述
CreateDot1XConfigurationRequest	请求信息包含： tt:Dot1XConfiguration Dot1XConfiguration[1][1]
CreateDot1XConfigurationResponse	应答信息是空信息
错误代码	原因分析
env:Receiver ter:ActionNotSupported ter:EAPMethodNotSupported	不支持建议的 EAP 方法设备
env:Receiver ter:Action ter:MaxDot1X	IEEE 802.1X 配置能够到达的最大数量
env:Sender ter:OperationProhibited ter:CertificateID	无效的证书 ID 错误
env:Sender ter:InvalidArgVal ter:ReferenceToken	Dot1XConfigurationToken 已经存在
env:Sender ter:InvalidArgVal ter:InvalidDot1X	无效的 IEEE 802.1X 配置

8.4.7.2 对 IEEE802.1X 配置

当 CreateDot1XConfiguration 命令试图创建一个新的参数配置时，通过这个操作可以修改设备已经存在的配置参数。支持 IEEE802.1X 的设备都应该支持这个命令。

表 66：SetDot1XConfigurationRequest 指令

SetDot1XConfiguration	请求与应答
信息名称	相应功能以及参数描述

SetDot1XConfigurationRequest	请求信息包含： tt:Dot1XConfiguration Dot1XConfiguration[1][1]
SetDot1XConfigurationResponse	应答信息是空信息
错误代码	原因分析
env:Receiver ter:ActionNotSupported ter:EAPMethodNotSupported	设备不支持建议的 EAP 方法
env:Sender ter:OperationProhibited ter:CertificateID	无效的 ID 错误
env:Sender ter:OperationProhibited ter:ReferenceToken	无效的 Dot1XConfiguration 令牌错误
env:Sender ter:InvalidArgVal ter:InvalidDot1X	无效的 IEEE802.1X 配置

8.4.7.3 获取 IEEE802.1X 配置

通过这个操作能够通过相关的令牌来获取 IEEE802.1X 配置参数。

如果一个设备支持 IEEE802.1x,那么设备就应该支持这个命令。

在检索配置时，无论 802.1x 方法是否有密码请求，设备都不应该在返回的响应中包含密码元素。

表 67：GetDot1XConfiguration 指令

GetDot1XConfiguration	请求与应答
信息名称	相应功能以及参数描述
GetDot1XConfigurationRequest	请求信息包含： tt:ReferenceToken Dot1XConfigurationToken[1][1]
GetDot1XConfigurationResponse	应答信息包含： tt:Dot1XConfiguration Dot1XConfiguration[1][1]
可能的存在的错误	原因分析
env:Sender ter:OperationProhibited ter:ReferenceToken	无效的 Dot1XConfigurationToke

8.4.7.4 获取 IEEE802.1X 配置

通过这个操作可以从设备中获取所有已经存在的 IEEE802.1X 设置。设备应该将对客户端应答其所有的 IEEE802.1X 配置，这样客户端才能知道有多少 IEEE802.1X 配置已经存在，以及它们是怎么被配置的。

一个支持 IEEE802.1X 的设备应该支持这个命令

无论在检索配置 802.1x 方法是否需要密码，设备在应答时候都不应该包含密码。

表 68：GetDot1XConfigurations 指令

GetDot1XConfigurations	请求与应答
信息名称	相应功能以及参数描述
GetDot1XConfigurationsRequest	请求信息是空信息
GetDot1XConfigurationsResponse	应答信息包含： tt:Dot1XConfigurationDot1XConfiguration[0][unbounded]
错误代码	原因分析
	没有与此命令相关的错误代码

8.4.7.5 删除 IEEE802.1X 配置

这个操作可以删除设备上的 IEEE802.1X 的配置参数。在请求的信息中会指定哪一个配置被删除。如果设备支持 IEEE802.1X，那么设备就应该支持这个命令。

表 69：DeleteDot1XConfigurations 命令

DeleteDot1XConfigurations	请求与应答
信息名称	相应功能以及参数描述
DeleteDot1XConfigurationRequest	请求信息包含任何： tt:ReferenceToken Dot1XConfigurationToken[1][1]
DeleteDot1XConfigurationResponse	应答信息是一条空信息
可能的存在的错误	原因分析
env:Sender ter:OperationProhibited ter:ReferenceToken	无效的 Dot1XConfigurationToken
env:Receiver ter:OperationProhibited ter:ReferenceToken	不能删除与 IEEE802.1x 相关的配置

8.4.8 创建签名证书

通过这个操作可以产生一个公钥/私钥对，在生成密钥对后会产生一个设备自签名证书。使用一个合适的板载密钥对产生机制来创建证书。

如果设备支持板载密钥对产生。支持 TLS 的设备应该支持证书创建指令。以及，如果设备支持板载密钥对的产生，那么支持 IEEE802.1X 协议的设备应该支持密钥对产生命令。通过证书 ID 来标识证书和密钥对，这个 ID 既可以被证书创建的申请者来选择，也可以通过设备来选择。

表 70: CreateCertificate 指令

CreateCertificate	请求与应答

信息名称	相应功能以及参数描述
CreateCertificateRequest	请求信息包含请求的证书 ID 以及其他的请求参数：主题，之前有效或者之后有效 xs:token CertificateID [0][1] xs:string Subject [0][1] xs:dateTime ValidNotBefore [0][1] xs:dateTime ValidNotAfter [0][1]
CreateCertificateResponse	应答信息包含自签名证书 tt:Certificate NvtCertificate [1][1]
可能的存在的错误	原因分析
env:Receiver ter:Action ter:KeyGeneration	产生私钥/公钥失败
env:Sender ter:InvalidArgVal ter:CertificateID	证书已经存在
env:Sender ter:InvalidArgVal ter:InvalidDateTime	ValidNotBefore or ValidNotAfter parameter 参数是无效的

8.4.9 获取证书

通过这个操作可以获取所有以 TLS 认证为目的服务器证书，以及所有以 IEEE802.1x 为目的的设备客户端证书。这个命令仅仅列举了设备的 TLS 服务端证书和 IEEE802.1X 客户端证书，证书以二进制数据进行返回；支持 TLS 的设备应该支持这个命令，以及证书应该通过[X.681], [X.682], [X.683] DER [X.690]编码规则进行编码

表 71：GetCertificates 指令

GetCertificates	请求与应答
信息名称	相应功能以及参数描述
GetCertificatesRequest	请求信息是一条空息
GetCertificatesResponse	应答信息包含设备证书清单 tt:Certificate NvtCertificate [0][unbounded]
错误代码	原因分析
	没有与此命令相关的错误代码

8.4.10 获取CA证书

在如下两种情况，设备能够对 CA 证书进行下载以及使用证书：

1.在 TLS 服务功能中，进行 TLS 客户认证；

2..在 IEEE802.1x 功能中，进行服务的认证。

这个操作能够获取所有下载进设备的 CA 证书。如果设备支持 TLS 客户认证或者 IEEE802.1x, 那么设备就应该支持这个命令，并返回一个通过[X.681], [X.682], [X.683] DER [X.690]编码规则的证书

表 72：GetCACertificates 指令

GetCertificates	请求与应答
信息名称	相应功能以及参数描述
GetCACertificatesRequest	请求信息是一条空信息
GetCACertificatesResponse	应答信息包含 CA 证书清单 tt:Certificate CACertificate [0][unbounded]
错误代码	原因分析
	没有与此命令相关的错误代码

8.4.11获取证书状态

此操作是特定于 TLS 功能。这个操作能够获取设备 TLS 服务证书的状态。如果一个设备支持 TLS，那么设备就应该支持这个命令

表 73：GetCertificatesStatus 指令

GetCertificatesStatus	请求与应答
信息名称	相应功能以及参数描述
GetCertificatesStatusRequest	请求信息是一条空信息
GetCertificatesStatusResponse	应答信息包含与设备服务证书，ID 相关状态清单，状态被定义为布尔型： tt:CertificateStatus CertificateStatus [0][unbounded]
错误代码	原因分析
	没有与此命令相关的错误代码

8.4.12 设置证书状态

此操作是只适用于 TLS 功能。通过这个操作可以设置设备 TLS 服务证书的状态，一个支持 TLS 的设备应该支持这个命令。一次只能使能一个设备服务证书。

表 74：SetCertificatesStatus 指令

SetCertificatesStatus	请求与应答

信息名称	相应功能以及参数描述
SetCertificatesStatusRequest	请求信息包含了与 ID 和请求证书状态相关的设备服务证书清单： tt:CertificateStatus CertificateStatus [0][unbounded]
SetCertificatesStatusResponse	应答信息不包含任何信息
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:CertificateID	请求信息包含的证书未知

8.4.13 获取证书请求

这个操作向设备请求一个 PCK#10 签名证书，这返回的信息要么被完全格式化为指定 PKCS ＃ 10]格式或者采用 PEM 编码为 PKCS ＃ 10 格式。为了让这个命令能够被执行，设备必须有一个私钥/与公钥对。这密钥对与输入的 CertificateID 的参数有关。这 CertificateID 是通过 CreateCertificate 命令才产生的。

如果一个设备支持板载密钥对的产生，也就是说设备支持通过 TLS 或者 IEEE802.1X 来产生密钥对，那么设备就支持这个命令。

表 75：GetPkcs10Request 指令

GetPkcs10Request	请求与应答
信息名称	相应功能以及参数描述
GetPkcs10RequestRequest	请求信息包含一个相关证书以及和这个证书相关的一些参数。这些属性被作为 DER ASN.1 目标进行编码 xs:token CertificateID [1][1] xs:string Subject [0][1] xs:BinaryData Attributes [0][1]
GetPkcs10RequestResponse	应答信息包含 PKCS#10 请求数据结构 tt:BinaryData Pkcs10Request [1][1]
可能的存在的错误	原因分析
env:Sender ter:InvalidArgVal ter:CertificateID	无效的 CertificateID
env:Receiver ter:Action ter:Signature	PKCS#10 签名创建失败

8.4.14 获取客户证书状态

这个操作是特定的 TLS 功能。通过这个操作可以获取设备 TLS 的客户认证状态。支持 TLS 的设备应该支持这个命令。

表 76：GetClientCertificateMode 指令

GetClientCertificateMode	请求与应答
信息名称	相应功能以及参数描述
GetClientCertificateMode-Request	请求信息是一条空信息
GetClientCertificateMode-Response	应答信息包含设备客户的认证状态： xs:boolean Enabled [1][1]
错误代码	原因分析
	没有与此命令相关的错误代码

8.4.15 设置客户认证状态

这个是特定的 TLS 功能。通过这个操作可以设置设备 TLS 客户认证状态。一个支持 TLS 的设备应该在支持这个命令。

表 77：SetClientCertificateMode 指令

SetClientCertificateMode	请求与应答
信息名称	相应功能以及参数描述
SetClientCertificateMode-Request	请求信息包含对设备客户认证状态的请求状态 xs:boolean Enabled [1][1]
SetClientCertificateMode-Response	应答信息是一条空信息
可能的存在的错误	原因分析
env:Receiver ter:InvalidArgVal ter:ClientAuth	使能客服认证，但是客户认证不被支持或者不支持配置

8.4.16 下载设备证书

通过使用 PCK#10 证书请求命令创建的 TLS 服务证书或者 IEEE802.1X 客户证书，这些证书能够通过命令来下载到设备中。在请求信息中应该包含证书 ID。设备可以根据证书的公钥以及主题信息对证书分类。

在请求信息中的证书 ID 是客户希望拥有的 ID。设备被假想来扫描存在于设备中钥匙对，进而来确定谁是下载的证书的密钥对，然后对证书和密钥对进行连接。

如果支持 TLS 或者 IEEE802.1X 的的板载密钥对产生的设备应该支持这命令。

证书应该通过 ASN.1 [X.681], [X.682], [X.683] DER [X.690]编码规则来进行编码。

尽管这参数由于历史原因被称为 NVTCertificate，但这个命令能够被用在任何设备类型。

表 78：LoadCertificates 命令

LoadCertificates	请求与应答
信息名称	相应功能以及参数描述

LoadCertificatesRequest	请求信息包含要上传的设备证书清单 tt:Certificate NVTCertificate [1][unbounded]
LoadCertificatesResponse	应答信息是空信息
可能的存在的错误	原因分析
env:Receiver ter:InvalidArgVal ter:ClientAuthenv:Sender ter:InvalidArgVal ter:CertificateFormat	证书格式不可用，或者设备不支持这证书格式
env:Sender ter:InvalidArgVal ter:CertificateID	Certificate ID 已经存在
env:Sender ter:InvalidArgVal ter:InvalidCertificate	无效的 证书

8.4.17 利用私有密钥来链接下载设备证书

这里可能存在关于证书授权或者一些其他一些没有 PKCS#10 证书签名的情况。在这些情况下，证书将与私钥进行捆绑。这个命令将在这样的情况下使用。在请求信息的证书 ID 是被选择性的赋予客户希望的值。如果在请求的信息中没有指定证书的 ID，那么设备可以根据具体情况选择 ID。

这个操作会输入一个私钥/公钥对进设备。

证书能够使用 ASN.1 [X.681], [X.682], [X.683] DER [X.690]编码规则进行编码。

如果设备不支持板载密钥对生成，以及不支持使用客户证书的 TLS 或者 IEEE802.1X，那么设备就应该支持这个命令。而支持板载密钥对的生成的设备可以支持这个命令。支持这个操作设备的安全策略应该确保私钥受到充当保护。

表 79：LoadCertificateWithPrivateKey 指令

LoadCertificateWithPrivateKey	请求与应答
信息名称	相应功能以及参数描述
LoadCertificateWithPrivateKeyRequest	请求信息包含要输入的私钥公钥对 tt:CertificateWithPrivateKey CertificateWithPrivateKey[1][unbounded]
LoadCertificateWithPrivateKeyResponse	应答信息是一条空信息
可能的存在的错误	原因分析
env: Sender ter:InvalidArgVal ter:CertificateFormat	证书格式不可用，或者设备不支持这证书格式

env:Sender ter:InvalidArgVal ter:CertificateID	Certificate ID 已经存在
env: Sender ter:InvalidArgVal ter: KeysNotMatching	公钥与私钥不匹配

8.4.18 获取证书信息请求

通过这个操作可以用来请求与指定 ID 相关的证书信息。设备应该响应“Issuer DN”, “Subject DN”, “Key usage”, "Extended key usage”, “Key Length”, “Version”, “Serial Number”, “Signature Algorithm” and “Validity”这样的证书信息，只要设备能够检索相关证书的这些信息。这 IssuerDN 和 SubjectDN 应该通过[RFC4514]的方式进行编码。

支持 TLS 或者 IEEE 802.1X 协议的设备应该都支持这个命令。

表 80：GetCertificateInformation 指令

GetCertificateInformation	请求与应答
信息名称	相应功能以及参数描述
GetCertificateInformationRequest	请求信息包含： CertificateID: The token of the certificate. xs: token CertificateID [1][1]
GetCertificateInformationResponse	应答信息包含 tt:CertificateInformation CertificateInformation[1][1]
可能的存在的错误	原因分析
env:Sender ter:InvalidArgVal ter:CertificateID	无效的证书 ID

8.4.19 下载CA证书

当为了进行确认 如：在 TLS 中客户服务的确认或者在 IEEE802.1X 功能中对服务证书的确认，这是很有必要下载一个信任的 CA 证书或者下载一个信任的根证书，这时候下载 CA 证书这个命令就会被用到。

支持 TLS 或者 IEEE802.1x 协议的设备支持 LoadCACertificates 这个命令，设备必须支持 DER 格式，而其他格式也可以被设备支持。设备应该可以根据公钥和主题信息来分类接收到的证书。要么是所有的 CA 证书被成功的下载或者返回一个没有下载任何 CA 证书的错误信息。

表 81：LoadCACertificates 指令

LoadCACertificates	请求与应答
信息名称	相应功能以及参数描述

LoadCACertificatesRequest	请求信息包含一个用于上传的 CA 证书设备清单 tt:Certificate CACertificate [1][unbounded]
LoadCACertificatesResponse	应答信息是一个空信息
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:CertificateFormat	格式错误或者设备不支持这种格式
env:Sender ter:InvalidArgVal ter:CACertificateID	CA 证书的 ID 已经存在
env:Receiver ter:OperationProhibited ter:MaxCertificates	超过了最大可下载的证书数量

8.4.20 删除证书

这个操作可以删除一个或者多个证书。设备也可以删除一个与证书相耦合的私钥/公钥对。如果设备支持 TLS 或者支持 IEEE802.1X，那么设备就应该支持通过 DeleteCertificates 删除一个或者多个证书。对于这个命令，要么是所有证书被删除，要么返回一个没有删除任何证书的信息。

表 82：DeleteCertificates 命令

DeleteCertificates	请求与应答
信息名称	相应功能以及参数描述
DeleteCertificatesRequest	请求信息包含一个删除证书的 CertificateIDparameter. xs:token CertificateID[1][unbounded
DeleteCertificatesResponse	应答信息是一条空信息
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:CertificateID	请求信息中包含的证书不可知
env:Receiver ter:OperationProhibited ter:CertificateID	不能够删除指定的证书

8.4.21 获取远程用户

通过这个操作可以返回远程用户的配置信息。支持远程用户处理的设备应该支持这个操作。此操作仅仅作为 WS-UserToken 类别或者 HTTP / RTSP 的用户是有效。

对于派生密码使用的算法在前面的 5.12.2 节已经描述了。

表 83：GetRemoteUser 指令

GetRemoteUser	请求与应答
信息名称	相应功能以及参数描述
GetRemoteUserRequest	请求信息是一条空信息
GetRemoteUserResponse	应答信息包含对远程用户的配置。这些值列举如下： xs:string Username [1][1] xs:boolean UseDerivedPassword [1][1] 注意：设备不应该放回远程用户的密码 tt:RemoteUser: RemoteUser [0][1]
代码错误	原因分析
env:Receiver ter:ActionNotSupported ter:NotRemoteUser	不支持远程用户处理

8.4.22 设置远程用户

这个操作能够设置远程用户。如果一个设备支持远程用户处理，那么设备就应该支持这个操作。这操作仅仅对于 WS-UserToken 类别的用户或者 HTTP / RTSP 用户是有效的。
这设置的密码应该是原密码而不是派生密码。

如果要设置派生密码，那么在连接到一个远程设备上时，进行求导。派生密码使用的具体算法在 5.12.2.1 节进行描述。

为了将移除用户，应该在对 SetRemoteUser 命令进行调用时不带 RemoteUser 参数。

表 84：SetRemoteUse 指令

SetRemoteUser	请求与应答
信息名称	相应功能以及参数描述
SetRemoteUserRequest	请求信息包含：远程用户；值如下： xs:string Username [1][1] xs:string Password [0][1] xs:boolean UseDerivedPassword [1][1] tt:RemoteUser: RemoteUser [0][1]
SetRemoteUserResponse	应答信息是一条空信息：

错误代码	原因分析
env:Receiver ter:ActionNotSupported ter:NotRemoteUser	不支持远程用户处理

8.4.23获取终端参数

一个客户端可以询问设备服务的终端相关的地址属性，利用这个属性可以用来衍生远程用户的密码，设备应该支持通过 GetEndpointReference 命令来获取设备服务终端相关地址服务属性。

表 85：GetEndpointReference 命令

GetEndpointReference	请求与应答
信息名称	相应功能以及参数描述
GetEndpointReferenceRequest	请求信息是一条空信息
GetEndpointReferenceResponse	应答信息包含请求的 URL xs:string GUID [1][1]
错误代码	原因分析
	没有与此命令相关的错误代码

8.5 输入与输出

这一节描述的命令与第九章相关

输入输出命令被用来控制状态或者观察 I/O 口的状态；如果设备拥有 I/0 口，那么设备支持 I/O 命令。

8.5.1获取继电器 输出

通过这个操作可以获得所有可用的继电器清单及他们配置。

表 86：GetRelayOutputs 命令

GetRelayOutputs	请求与应答
信息名称	相应功能以及参数描述
GetRelayOutputsRequest	请求信息不包含任何参数
GetRelayOutputsResponse	应答信息包含一个继电器输出矩阵： tt:RelayOutput RelayOutputs [0][unbounded]
错误代码	原因分析
	没有与此命令相关的错误代码

8.5.2 对继电器输出进行配置

这个操作用于对继电器输出进行配置
继电器可以工作在以下两种模式：

1.Bistable – 如果设置这种状态，继电器将保持在这种状态.

2.Monostable – 如果继电器设置这种状态，那么继电器将在规定时间内返回其空闲状态。

继电器输出的物理理想状态能够通过设置 IdleState 进行 open 或者 close 来进行配置

空闲状态 open 意味着当通过触发命令来设置继电器的状态成 inactive/active 来打开/关闭继电器。

空闲状态 close 意味着通过触发命令来将状态设置为‘active/inactive 来关闭/打开继电器。

表 87：SetRelayOutputSettings 命令

SetRelayOutputSettings	请求与应答
信息名称	相应功能以及参数描述
SetRelayOutputSettingsRequest	请求信息包含： “RelayToken”：与令牌环相关的继电器输出 “RelayOutputSettings”：设置的延时 tt:ReferenceToken RelayOutputToken [1][1] tt:RelayOutputSettings RelayOutputSettings [1][1]
SetRelayOutputSettingsResponse	应答信息是一个空信息
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:RelayToken	相关的令牌未知
env:Sender ter:InvalidArgVal ter:ModeError	单稳态延时时间无效

8.5.3继电器触发输出

通过这个操作可以触发一个继电器输出。

表 88：SetRelayOutputState 指令

SetRelayOutputState	请求与应答
信息名称	相应功能以及参数描述
SetRelayOutputStateRequest	请求信息包含： RelayToken”：与继电器输出相关的命令. “LogicalState”：触发请求 i.e., active or inactivet tt:ReferenceToken RelayOutputToken [1][1] tt:RelayLogicalState LogicalState [1][1]

SetRelayOutputStateResponse	应答信息是一个空信息
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:RelayToken	相关继电器令牌未知

8.5.4 辅助操作

这一节描述的操作是用来管理设备支持的辅助命令，控制与设备相关的设备，比如：一个红外线灯，一个加热器或者一个温度计。

可以从 GetCapabilities 命令的响应中检索到辅助的参数。通过这类命令转换的命令应该与响应的辅助数据的命令清单中的一种命令相匹配。如果命令响应的清单上只有 irlampon 命令，那么这 SendAuxiliary 命令的参数也将会是 irlampon，这也可以表明链接的 IR 灯被打开。

支持辅助服务能力的设备应该支持这个命令。

表 89：Send auxiliary 命令

SendAuxiliaryCommand	请求与应答
信息名称	相应功能以及参数描述
SendAuxiliaryCommandRequest	请求信息包含辅助命令： tt:AuxiliaryData AuxiliaryCommand[1][1]
SendAuxiliaryCommandResponse	应答信息是一个辅助的应答 tt:AuxiliaryData AuxiliaryCommandResponse[0][1]
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:AuxiliaryDataNotSupported	请求信息的辅助命名不支持

8.6 与服务相关的错误代码

表 90 列举与设备服务相关的错误代码，另外每一个命令也能产生相似错误，参见表 6.

表 90：一类错误定义成一个子代码，在这些子代码下是具体的错误。

表 90：Device service specific fault codes

错误代码	父类子码 子码	错误原因	描述
env:Receiver	ter:Action ter:EmptyPolicy	策略是空的	设备策略文件不存在或者是空的

env:Receiver	ter:Action ter:EmptyScope	范围清单是空的	范围清单是空的
env:Receiver	ter:Action ter:FirmwareUpgradeFailed	升级失败	固件升级 失败
env:Receiver	ter:Action ter:KeyGeneration	生成密钥失败	私钥/公钥生成失败。
env:Receiver	ter:Action ter:Signature	创建一个签名失败。	PKCS#10 签名创建失败
env:Receiver	ter:InvalidArgVal ter:ClientAuth	不支持客户认证	使能客户认证，但是客户认证不被支持或者没有配置
env:Receiver	ter:Action ter:TooManyUsers	太多使用者	超出最大使用者的数量
env:Receiver	ter:Action ter:TooManyScopes	范围清单太大	范围清单超出了支持的范围数量
env:Receiver	ter:ActionNotSupported ter:NoSuchService	不支持这样的服务	设备不支持这请求的 WDSL 服务类别
env:Sender	ter: InvalidArgs ter:AccesslogUnavailable	没有日志可供访问	这里不存在可供访问的日志信息
env:Sender	ter:InvalidArgVal ter:CertificateFormat	无效的格式	无效的证书格式或者设备不支持这样的格式
env:Sender	ter:InvalidArgVa ter:CertificateID	无效的证书 ID	不能识别提到证书或者证书的 ID 已经存在
env:Sender	ter:InvalidArgVal ter:CACertificateID	无效的证书 ID	CA 证书不被识别或者 CA 证书参数已经存在
env:Sender	ter:InvalidArgVal ter:InvalidBackupFile	无效的文件	备份的文件无效
env:Sender	ter:InvalidArgVal ter:InvalidDateTime	无效的日期和时间	指定的是一个无效的日期和时间
env:Sender	ter:InvalidArgVal	无效的名字	建议的 NTP 服务名字是无

	ter:InvalidDnsName		效的
env:Sender	ter: InvalidArgs ter:InvalidFirmware	无效固件	固件无效；不被设备支持
env:Sender	ter:InvalidArgVal ter:InvalidGatewayAddress	无效的地址	提供的网关地址无效
env:Sender	ter:InvalidArgVal ter:InvalidHostname	无效的名字	设备不能够接受请求信息中的设置的主机名
env:Sender	ter:InvalidArgVal ter:InvalidInterfaceSpeed	无效的速度	建议的速度不被支持
env:Sender	ter:InvalidArgVal ter:InvalidInterfaceType	无效的类型	建议的网络接口类型不被支持
env:Sender	ter:InvalidArgVal ter:InvalidIPv4Address	无效的地址	建议设置的 IPV4 地址是无效的
env:Sender	ter:InvalidArgVal ter:NoIPv4Address	地址不存在	IPV4 地址被移除了或者不存在
env:Sender	ter:InvalidArgVal er:InvalidIPv6Address	无效的地址	建议设置的 IPV6 地址是无效的
env:Sender	ter:InvalidArgVal ter:NoIPv6Address	地址不存在	IPV6 地址被移除或者根本不存在
env:Sender	ter:InvalidArgVal ter:InvalidMtuValue	无效的数据	MTU 值无效
env:Sender	ter:InvalidArgVal ter:InvalidNetworkInterface	无效的令牌环	支持的网络接口令牌环不存在
env:Sender	ter:InvalidArgVal ter:InvalidTimeZone	无效的数据	指定的是一个无效的时间区域
env:Sender	ter:InvalidArgVal ter:IPFilterListIsFull	清单满了	由于清单满了，不能够添加 IP 滤波器了

env:Sender	ter:InvalidArgVal ter:ModeError	无效数据	单稳延时无效
env:Sender	ter:InvalidArgs ter:PolicyFormat	无效格式	由于不知道策略的格式，请求的策略不能够设置
env:Sender	ter:InvalidArgVal ter:RelayToken	无法识别继电器的令牌	参考的令牌无法识别
env:Sender	ter:InvalidArgVal ter:ServiceNotSupported	不支持这种服务	提供的网络服务不被支持
env:Sender	ter:InvalidArgVal ter:SupportInformationUnavailable	没有可用的支持信息	不存在支持的可用信息
env:Sender	ter: InvalidArgs ter:SystemlogUnavailable	没有可用的系统日志	不存在可用的日志信息
env:Sender	ter:InvalidArgVal ter:UsernameMissing	不能识别使用者的名字	不能识别使用者的名字
env:Sender	ter:OperationProhibited ter:FixedScope	企图删除固定的范围参数	企图删除固定的范围参数，产生命令排斥
env:Sender	ter:InvalidArgVal ter:NoScope	范围不存在	试图删除不存在的范围
env:Sender	ter:OperationProhibited ter:Password	密码太弱	密码太弱
env:Sender	ter:OperationProhibited ter:PasswordTooLong	密码太长	密码太长
env:Sender	ter:OperationProhibited ter:UsernameTooShort	太长的密码	密码太短
env:Sender	ter:OperationProhibited ter:ScopeOverwrite	试图修改永久设备	范围参数试图修改永久设备的范围设置
env:Sender	ter:OperationProhibited	用户名已经存在	用户名已经存在

	ter:UsernameClash		
env:Sender	ter:OperationProhibited ter:UsernameTooLong	用户名太长	用户 名太长
env:Sender	ter:ActionNotSupported ter:InvalidDot11	不支持	IEEE 802.11 的配置不被支持
env:Sender	ter:InvalidArgVal ter:InvalidSecurityMode	不支持	选择的安全模式不被支持
env:Sender	ter:InvalidArgVal ter:InvalidStationMode	不支持	选择的基站模式不被支持
env:Sender	ter:InvalidArgVal ter:MissingDot11	IEEE 802.11 值丢失	在安全配置中， IEEE 802.11 值丢失
env:Sender	ter:InvalidArgVal ter:MissingPSK	PSK 值丢失	在安全配置中 PSK 的值丢失
env:Sender	ter:InvalidArgVal ter:MissingDot1X	IEEE802.1X 值丢失	IEEE802.1 的值在安全配置中丢失或者不存在
env:Sender	ter:InvalidArgVal ter:IncompatibleDot1X	IEEE802.1X 值不兼容	IEEE 802.1X 的值在与网络接口的安全配置中不兼容
env:Sender	ter:InvalidArgVal ter:NotDot11	NotIEEE 802.11	接口不是 IEEE 802.11 接口
env:Sender	ter:InvalidArgVal ter:InvalidDot1X	无效的 IEEE802.11 配置	与 IEEE 802.11 的相关配置是无效的
env:Sender	ter:Action ter:NotConnectedDot11	IEEE 802.11 没有连接	IEEE 802.11 网络没有连接
env:Sender	ter:ActionNotSupported ter:NotScanAvailable	不支持扫描可用的 IEEE 802.11 网络	不支持扫描可用的 IEEE 802.11 网络
env:Sender	ter:ActionNotSupported ter:NotRemoteUser	不支持远程用户处理	远程用户处理不被支持
env:Sender	ter:ActionNotSupported	建议的 EAP 方法	建议的 EAP 方法不被支持

	ter:EAPMethodNotSupported	不被支持	不被支持
env:Sender	ter:Action ter:MaxDot1X	已经达到最大的 IEEE 802.1X 配置数量	已经到达设备最大的 IEEE 802.1X 配置数量
env:Sender	ter:OperationProhibited ter:ReferenceToken	不能够删除相关的 IEEE 802.1X 配置	不可能删除与 IEEE 802.1X 相关的配置
env:Sender	ter:OperationProhibited ter:CertificateID	不能删除相关的证书	不可能删除相关的证书
env:Sender	ter:OperationProhibited ter:ReferenceToken	无效的 DOTX 令牌环配置	相关的证书 ID 是无效的
env:Sender	ter:OperationProhibited ter:CertificateID	无效的证书 ID	相关的证书 ID 是无效的
env:Sender	ter:InvalidArgVal ter:ReferenceToken	Dot1XConfigurationToken 已经存在	相关的 Dot1XConfigurationToken 在设备中已经存在
env:Sender	ter:InvalidArgVal ter:InvalidCertificate	无效的证书	相关的证书是无效的
env:Sender	ter:OperationProhibited ter:MaxCertificates	已经达到允许下载的最大证书数量	已经到达设备的最大证书下载数量
env:Sender	ter:OperationProhibited ter:PasswordTooWeak	密码太弱	密码太弱
env:Sender	ter:InvalidArgVal ter:AuxiliaryDataNotSupported	请求的辅助命令不被支持；	不支持请求的辅助命令。
env:Sender	ter:InvalidArgVal ter:InvalidTimeOutValue	相关的 Timeout 值无效	相关的 Timeout 值无效
env:Sender	ter:OperationProhibited ter:DataLengthOver	超出可用的字节数	超出可用的字节数
env:Sender	ter:OperationProhibited ter:DelimiterNotSupport	不支持的字符序列或者分隔符	不支持的字符序列或者分隔符
env:Sender	ter:OperationProhibited ter:InvalidMode	对于操作命令设备没有准备	对于操作命令，设备没有准备

env:Sender	ter:InvalidArgVal ter:FixedUser	移除固有的使用者	客户端试图删除固有的用户
env:Sender	ter:OperationProhibited ter:AnonymousNotAllowed	用户等级没被定义	用户等级没定义
env:Sender	ter:InvalidArgVal ter:KeysNotMatching	密码不匹配	公钥与私钥不匹配

9 设备 IO 服务

这个服务提供的命令集可以用来检索和配置设备的输入和输出端口。
请求设备返回有效的音视频输入输出端口和返回有效的继电器的命令定义是一样的，这个服务还提供获取和修改音视频输入输出端口配置信息的功能。

有输入源和输出端口的设备必须支持在[DeviceIOService.wsdl]中描述的服务。
这个服务定义的功能，有一部分同媒体服务中定义的重叠，如果一个设备（例如一个 NVT 设备）需要同时实现这两个服务，那些重叠部分的功能实现必须以这个服务为准，通过该服务中定义的命令集去实现音频的输入输出或视频源的配置。

9.1 视频输出

视频输出类型表示设备的视频输出接口，即用于连接显示器并输出视频信号的接口。该结构包含使用显示服务配置的布局设置（见第 14 章）。

9.1.1 获取视频输出集

该命令列出设备所有有效的视频输出端口。有一个或多个视频输出端口的设备支持通过 GetVideoOutputs 命令列出的有效的视频输出。

表 91：GetVideoOutputs 命令

GetVideoOutputs	请求-响应
消息名称	描述
GetVideoOutputsRequest	这是一个空消息。
GetVideoOutputsResponse	包含描述设备所有有效视频输出的结构清单。如果一个设备没有视频输出则返回一个空清单。 tt:VideoOutput VideoOutputs [0][unbounded]
错误码	描述
没有具体的错误代码。	

9.2 视频输出配置

9.2.1 获取视频输出配置

此操作请求视频输出配置。有一个或多个视频输出端口的设备支持通过此命令检索视频输出配置。

表 92：GetVideoOutputConfiguration 命令

GetVideoOutputConfiguration	请求-响应
消息名称	描述
GetVideoOutputConfiguration-Request	此消息包含视频输出令牌。 tt:ReferenceToken VideoOutputToken [1][1]
GetVideoOutputConfiguration-Response	此消息包含了与请求信息中的令牌相对应视频输出配置。 tt:VideoOutputConfiguration VideoOutputConfiguration [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoVideoOutput	请求信息中令牌指定的视频输出配置不存在。

9.2.2 设置视频输出配置

此操作修改视频输出配置。有一个或多个视频输出端口的设备支持通过这个命令设置其视频输出配置。

表 93：SetVideoOutputConfiguration 命令

SetVideoOutputConfiguration	请求-响应
消息名称	描述
SetVideoOutputConfiguration-Request	配置元素包含要修改的视频输出配置。 ForcePersistence 元素决定是否在重新启动后存储配置更改和保持。如果为真，改变是永久的。如果为假，可能会在重启后恢复到变化以前的值。 tt:VideoOutputConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]
SetVideoOutputConfiguration-Response	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoVideoOutput	要求的视频输出不存在。
env:Sender ter:InvalidArgVal ter:ConfigModify	配置参数无法设置。

9.2.3 获取视频输出配置选项集

此操作请求视频输出的视频输出配置选项集。有一个或多个视频输出的设备支持通过此命令检索视频输出配置选项集。

表 94：GetVideoOutputConfigurationOptions 命令

GetVideoOutputConfigurationOptions	请求-响应
消息名称	描述

GetVideoOutputConfiguration-OptionsRequest	VideoOutputToken 元素指定目标选项，设备中必须存在视频输出。 tt:ReferenceToken VideoOutputToken [1][1]
GetVideoOutputConfiguration-OptionsRequest	设备响应视频输出选项集。 tt:VideoOutputConfigurationOptions VideoOutputOptions [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoVideoOutput	请求视频输出不存在。

9.3 视频源

视频源代表视频输入端口。结构中包含视频像素分辨率，帧率和成像设置。例如，成像设置可以通过成像服务对对焦、曝光、亮度的参数进行操纵。

9.3.1 获取视频源

此操作列出了设备所有有效的视频源。有一个或多个视频输入端口的设备支持通过 GetVideoSources 命令列出有效的视频源。

表 95：GetVideoSources 命令

GetVideoSources	请求-响应
消息名称	描述
GetVideoSourcesRequest	空消息
GetVideoSourcesResponse	包含了设备所有描述有效视频源的结构清单。如果一个设备没有视频源，则返回一个空清单 tt:VideoSource VideoSource [0][unbounded]

错误码	描述
没有具体的错误代码。	

9.4 视频源配置

视频源配置包含一个相关视频源和描写视频边界的结构，该结构可以包含整个视频源的全部或者一部分像素面积。结构和视频源定义了流向客户的图像。

9.4.1 获取视频源配置

此操作列出了视频源的视频源配置。有一个或多个视频源的设备支持 GetVideoSourceConfigurations 命令。

表 96：GetVideoSourceConfiguration 命令

GetVideoSourceConfiguration	请求-响应
消息名称	描述
GetVideoSourceConfiguration-Request	此消息包含视频输入的令牌。 tt:ReferenceToken VideoSourceToken [1][1]
GetVideoSourceConfiguration-Response	此消息包含请求匹配令牌的视频源配置。 tt:VideoSourceConfiguration VideoSourceConfiguration [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoVideoSource	请求信息中视频令牌指定的视频源不存在。

9.4.2 设置视频源配置

此操作修改一个视频输入配置。有一个或多个视频源的设备通过这个命令支持设置视频源配置。

表 97：SetVideoSourceConfiguration 命令

SetVideoSourceConfiguration	请求-响应
消息名称	描述
SetVideoSourceConfiguration-Request	Configuration 元素包含要修改的视频信号源设置。配置包含一个指定的要修改的视频源设置元素。视频源必须存在于设备中。 ForcePersistence 元素决定是否在重启后存储和保持配置更改。如果为真，改变是永久的。如果为假，可能会在重新启动后恢复到改变以前的值。 tt:VideoSourceConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]
SetVideoSourceConfiguration-Response	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoVideoSource	请求的视频源不存在。
env:Sender ter:InvalidArgVal ter:ConfigModify	配置参数无法设置。

9.4.3 获取视频源多个配置选项

此操作请求视频源多个配置选项。有一个或多个视频源的设备支持此命令。

表 98：GetVideoSourceConfiguartionOptions 命令

GetVideoSourceConfiguartionOptions	请求-响应
消息名称	描述
GetVideoSourceConfiguration-OptionsRequest	视频源令牌元素指定目标视频输入选项。视频输入存在于设备中。 tt:ReferenceToken VideoSourceToken[1][1]
GetVideoSourceConfiguartion-OptionsResponse	VideoSourceOptions 返回有效的界以及提供给视频源令牌有效的元素。此栏设置请求的源的选项。 tt:VideoSourceConfigurationOptions VideoSourceOptions [1][1]

错误码	描述
env:Sender ter:InvalidArgVal ter:NoVideoSource	请求的视频输入不存在。

9.5 音频输出

音频输出表示可以连接到扬声器的音频输出端口。

9.5.1 获取多个音频输出

此命令列出所有有效的音频输出设备。有一个或多个音频输出端口的设备支持通过 GetAudioOutputs 命令列出有效的音频输出。

表 99：GetAudioOutputs 命令

GetAudioOutputs	请求-响应
消息名称	描述
GetAudioOutputsRequest	空消息。
GetAudioOutputsResponse	包含一个描述所有有效的音频输出设备的结构清单。如果设备没有音频输出，则返回一个空清单。 tt:AudioOutput AudioOutputs [0][unbounded]
错误码	描述
env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	设备不支持音频或音频输出。

9.6 音频输出配置

音频输出配置包含了现有的音频输出的参考。音频输出配置包含一个参数来控制输出电平。

9.6.1 获取音频输出配置

此操作请求音频输出配置。有一个或多个音频输出的设备支持通过此命令检索音频输出配置。

表 100：GetAudioOutputConfiguration 命令

GetAudioOutputConfiguration	请求-响应
消息名称	描述
GetAudioOutputConfigurationRequest	此消息包含音频输出令牌。 tt:ReferenceToken AudioOutputToken [1][1]
GetAudioOutputConfigurationResponse	此消息包含请求与令牌匹配的音频输出配置。 tt:AudioOutputConfiguration AudioOutputConfiguration [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoAudioOutput	指示请求的音频输出不存在音频输出令牌。

9.6.2 设置音频输出配置

此操作修改音频输出配置。有一个或多个音频输出的设备通过这个命令设置音频输出配置。

表 101：SetAudioOutputConfiguration 命令

SetAudioOutputConfiguration	请求-响应
消息名称	描述
SetAudioOutputConfiguration -Request	Configuration 元素包含修改后的音频输出配置。配置包含一个存在于设备中的指定要修改其配置的音频输出元素。 ForcePersistence 元素决定是否在重启后存储和保持配置更改。如果为真，改变是永久的。如果为假，可能会在重新启动后恢复到改变以前的值。 tt:AudioOutputConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]
SetAudioOutputConfiguration -Response	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoAudioOutput	所请求的音频输出不存在。
env:Sender ter:InvalidArgVal ter:ConfigModify	配置参数无法设置。

9.6.3 获取音频输出多个配置选项

此操作请求音频输出多个配置选项。有一个或多个音频输出的设备支持此命令。

表 102：GetAudioOutputConfigurationOptions 命令

GetAudioOutputConfigurationOptions	请求-响应
消息名称	描述
GetAudioOutputConfiguration-OptionsRequest	AudioOutputToken 元素选择指定的存在于设备中的音频输出。 tt:ReferenceToken AudioOutputToken[1][1]
GetAudioOutputConfiguration-OptionsResponse	AudioOutputsOptions 返回有效值范围为发送优先级和输出电平与音频输出令牌一样。此字段为设置请求源的输出选项。 tt:AudioOutputConfigurationOptions AudioOutputOptions [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoAudioOutput	所请求的音频输出不存在。

9.7 音频源

音频源表示未编码的音频输入和输入通道的数量状态。

9.7.1 获取音频源

此操作列出了所有有效的音频源设备。有一个或多个音频源设备通过 GetAudioSources 命令列出有效的音频输入清单。

表 103：GetAudioSources 命令

GetAudioSources	请求-响应
消息名称	描述
GetAudioSourcesRequest	空消息。

GetAudioSourcesResponse	包含一个描述设备所有有效的音频源的结构清单。如果一个设备没有音频输入，则返回一个空清单。 tt:AudioSource AudioSource [0][unbounded]
错误码	描述
env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	NVT 不支持音频。

9.8 音频源配置

音频源配置包含了音频信号源的参考。

9.8.1 获取音频源配置

此操作列出了一个音频输入配置。有一个或多个音频输入的设备支持 GetAudioSourceConfiguration 命令。

表 104：GetAudioSourceConfiguration 命令

GetAudioSourceConfiguration	请求-响应
消息名称	描述
GetAudioSourceConfiguration-Request	此消息包含音频源令牌。 tt:ReferenceToken AudioSourceToken [1][1]
GetAudioSourceConfiguration-Response	此消息包含请求与令牌匹配的音频源配置。 tt:AudioSourceConfiguration AudioSourceConfiguration [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoAudioSource	请求的音频源令牌 AudioSourceToken 不存在。

9.8.2 设置音频源配置

此操作修改音频源的配置。有一个或多个音频源的设备支持通过这个命令设置音频源配置。

表 105：SetAudioSourceConfiguration 命令

SetAudioSourceConfiguration	请求-响应
消息名称	描述
SetAudioSourceConfiguration-Request	配置元素包含修改后的音频源配置。配置包含一个存在设备中的要修改音频源的配置元素。 ForcePersistence 元素决定是否在重启后存储和保持配置更改。如果为真，改变是永久的。如果为假，可能会在重新启动后恢复到改变以前的值。 tt:AudioSourceConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]
SetAudioSourceConfiguration-Response	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoAudioSource	请求的音频源不存在。
env:Sender ter:InvalidArgVal ter:NoAudioSource	配置参数无法设置。

9.8.3 获取音频源多个配置选项

此操作请求音频源多个配置选项。有一个或多个音频源的设备支持此命令。

表 106：GetAudioSourceConfigurationOptions 命令

GetAudioSourceConfigurationOptions	请求-响应
消息名称	描述
GetAudioSourceConfiguration-OptionsRequest	AudioSourceToken 元素指定其请求的音频输入选项。音频源应存在于设备中。 tt:ReferenceToken AudioSourceToken[1][1]
GetAudioSourceConfiguration-Response	音源选项返回有效的音频源令牌。这里应该设置成请求选项的源。 tt:AudioSourceConfigurationOptionsAudioSourceOptions [1][1]

错误码	描述
env:Sender ter:InvalidArgVal ter:NoAudioSource	请求音频输入不存在。

9.9 继电器输出

输入/输出（ I / O）命令是用来控制或观察 I / O 端口的状态。如果设备有 I / O 端口，那么就应支持 I / O 命令。

继电器输出也在“设备管理”被定义 （见输入/输出（ I / O））。继电器输出可以同时访问设备管理服务和设备的 IO。

9.9.1 获取多个继电器输出

此操作获取所有有效继电器输出和设置清单。

表 107：GetRelayOutputs 命令

GetRelayOutputs	请求-响应
消息名称	描述
GetRelayOutputsRequest	空消息。
GetRelayOutputsRequest	此消息包含一个继电器输出数组。 tt:RelayOutput RelayOutputs [0][unbounded]
错误码	描述
	没有具体的错误命令！

9.9.2 设置继电器输出设置

此操作设置一个继电器输出的配置。

继电器可以工作在两个中继模式：

双稳态 - 设置状态后，继电器保持在这种状态下。

单稳态 - 设置状态后，继电器在设定的时间后返回到闲置状态。

可设置继电器输出端口闲置状态的配置为“开启”或“关闭” （反转继电器行为）。

闲置状态“开启”是将继电器通过触发命令设置继电器状态为“无效”（见 8.5.3 节），通过相同的命令设置“启动”则是关闭状态。

闲置状态'关闭' 是继电器闭合时将继电器状态通过触发命令设置为 “无效” （见 8.5.3 节），通过相同的命令设置“启动”则是开放状态。

表 108：SetRelayOutputSettings 命令

SetRelayOutputSettings	请求-响应
消息名称	描述
SetRelayOutputSettingsRequest	此消息包含： “ RelayOutputToken ” ： 继电器输出的参考令牌。 “ RelayOutputSettings ” ： 继电器设置集。 tt:ReferenceToken RelayOutputToken [1][1] tt:RelayOutputSettings RelayOutputSettings [1][1]
SetRelayOutputSettingsResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:RelayToken	未知继电器参考令牌。
env:Sender ter:InvalidArgVal ter:ModeError	单稳延时时间无效

9.9.3 触发继电器输出

此操作触发继电器输出2。

表 109：SetRelayOutputState 命令

SetRelayOutputState	请求-响应
消息名称	描述
SetRelayOutputStateRequest	此消息包含： “RelayOutputToken “ ：令牌要求继电器输出。 “ LogicalState ”：触发请求，即有效或无效。 tt:ReferenceToken RelayOutputToken [1][1] tt:RelayLogicalState LogicalState [1][1]
SetRelayOutputStateResponse	空消息。

错误码	描述
env:Sender ter:InvalidArgVal ter:RelayToken	未知继电器参考令牌。

[2]没有 GetRelayState 命令;继电器输出电流的逻辑状态是通过发送通知和它们的属性确定的。

9.10服务错误码

表 110 列出设备 IO 服务特定的错误代码。此外，每个命令还可以导致一个通用的错误，见表 6 。

表 110：设备 IO 服务的具体错误码

错误码	母码 子码	错误原因	描述
env:Sender	ter:InvalidArgVal ter:ConfigModify	无效的配置参数	配置参数无法设置。
env:Sender	ter:InvalidArgVal ter:NoVideoOutput	视频输出令牌不存在。	要求的视频输出标示与视频输出令牌不存在。
env:Sender	ter:InvalidArgVal ter:NoVideoSource	视频源令牌不存在。	要求的视频源标示与视频源令牌不存在。
env:Sender	ter:InvalidArgVal ter:NoAudioOutput	音频输出令牌不存在。	要求的音频输出标示与音频输出令牌不存在。
env:Sender	ter:InvalidArgVal ter:NoAudioSource	音频源令牌不存在。	要求的音频源标示与音频源令牌不存在。
env:Sender	ter:InvalidArgVal ter:RelayToken	未知的继电器参考。	要求的继电器输出标示与继电器输出令牌不存在。
env:Sender	ter:InvalidArgVal ter:ModeError	单稳延时时间不正确	

10 图像配置

图像服务提供对设备的图像性能进行控制和配置的操作。有一个或多个视频源的设备支持 [ONVIF 的影像 WSDL]中定义的图像服务。图像设置是视频源实体的一部分。也就是说，图像参数直接影响一个指定的视频源。

10.1 图像设置

图像服务提供操作来获取或设置图像参数和这些参数的有效范围。有些参数如果没有在一个特定的模式中设置则是无效的。在图像配置中的一些参数需要特定的图像功能支持，可以通过 GetOptions 命令获取图像功能信息。以下设置都可以通过影图像服务操作：

BacklightCompensation ： 开启/关闭背光补偿模式（开/关）。

- On
  - O 可选的级别参数（无单位）。
- Off

Brightness :调整图像的亮度（无单位）。

ColorSaturation ： 调整图像的色彩饱和度（无单位）。

Sharpness ： 调整图像的清晰度（无单位）。

Contrast ： 调整图像的对比度（无单位）。

Exposure(曝光)：

- Auto – 启用设备上的曝光算法：
  - O Priority - 设置优先曝光模式（低噪音/帧率）。
  - O Window -矩形曝光掩模。
  - O Min/ MaxExposureTime - 曝光时间范围内允许被使用的算法。
  - O Min / MaxGain - 传感器增益范围，允许所使用的算法。
  - O Min / MaxIris - 光圈范围允许被使用的算法。
- Manual – 禁用设备上的曝光算法：
  - O ExposureTime -图像传感器使用固定的曝光时间（μs）。
  - O Gain - 图像传感器所使用的固定增益（dB）。

- Iris – 光圈影响的输入光固定衰减（dB）。0dB 图为一个完全开放光圈。

Focus(焦点):

- Auto （参数，适用于自动模式）：
  - Near/FarLimit - 焦镜头的限度(m)。
- Manual（适用于手动模式的参数）：
  - Default speed - 焦点移动操作的默认速度（当速度参数不存在时）。通过移动命令实现手动控制，请参阅第 10.1.4 节。

Ir cut filter(红外滤光片)：在开、关和自动之间切换红外截止滤波器的状态。自动状态让曝光算法处理的红外滤光片打开或关闭时机。

Whitebalance(白平衡):

- 自动白平衡模式(自动/手动)。
- 手动(适用于手动模式的参数)：
  - Rgain (无量纲)
  - Bgain (无量纲)

WideDynamicRange：宽动态范围（开/关）：

- On
  - 可选的级别参数（无量纲）。
- Off

有效的图像设置可以通过媒体服务的一部分--GetVideoSources 命令检索，指定在第 11.3.1 节。图像设置是视频源的一部分。

10.1.1 获取图像设置

此操作请求设备上的视频源的图像设置。如果视频源支持任何在[ONVIF 的架构]定义的图像设置类型，那么它可以通过 GetImagingSettings 命令从设备检索到图像设置。

图像设置参数参照 10.1 节中的描述。

表 111：GetImagingSettings 命令

GetImagingSettings	请求-响应
消息名称	描述
GetImagingSettingsRequest	此消息包含成像设置集请求的视频源的参考。 tt:ReferenceToken VideoSourceToken[1][1]
GetImagingSettingsResponse	此消息包含视频源请求的成像设置集。 tt:ImagingSettings20 ImagingSettings [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoSource	请求的视频源不存在。
env:Receiver ter:ActionNotSupported ter:NoImagingForSource	请求的视频源不支持图像设置。

10.1.2 设置图像设置

此操作设置设备上的视频源的图像设置。如果设备支持的任何成像类型在[ONVIF 的架构]中被定义，那么它可以通过 SetImagingSettings 命令在设备中配置这些参数。

在 10.1 节描述可配置的图像设置参数。通过在 10.1.3 节中定义的命令设置选项。

表 112：SetImagingSettings 命令

SetImagingSettings	请求-响应
消息名称	描述
SetImagingSettingsRequest	此消息包含一个被设置的视频源和成像设置集的参考。 ForcePersistence 元素决定是否在重启后存储和保持配置更改。如果为真，改变是永久的。如果为假，可能会在重新启动后恢复到改变以前的值。 tt:ReferenceToken VideoSourceToken[1][1] tt:ImagingSettings20 ImagingSettings [1][1] xs:boolean ForcePersistence [0][1]
SetImagingSettingsResponse	此消息不包含任何响应。
错误码	描述
env:Sender ter:InvalidArgVa ter:NoSource	请求的视频源不存在。

env:Receiver ter:ActionNotSupported ter:NoImagingForSource	请求的视频源不支持图像设置。
env:Sender ter:InvalidArgVal ter:SettingsInvalid	请求的设置不正确。

10.1.3 获取选项

此操作得到设备图像参数的具体有效范围。如果设备支持 SetImagingSettings 命令设置设备上的图像参数，则应该支持通过 GetOptions 命令得到配置选项。

表 113：GetOptions 命令

GetOptions	请求-响应
消息名称	描述
GetOptionsRequest	请求的视频源成像参数选项参考。 tt:ReferenceToken VideoSourceToken[1][1]
GetOptionsResponse	此消息包含分类成像设备的具体参数的有效范围。 tt:ImagingOptions20 ImagingOptions [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoSource	请求的 VideoSource 不存在。
env:Receiver ter:ActionNotSupported ter:NoImagingForSource	请求的 VideoSource 不支持图像设置。

10.1.4 移动

Move 命令以绝对，相对或连续的方式从当前位置移动对焦镜头。速度参数必须是连续的、可选择用绝对值或相对值控制。如果没有速度参数，则使用默认速度。通过焦距调整操作来关闭自动对焦。支持远程焦距控制的设备支持通过 Move 操作进行绝对、相对或连续控制。

图像功能指出了这个操作所支持具体操作。此操作需要至少有一个焦距控制性能起作用。

Move 操作包含以下命令：

Absolute – 规定位置参数和可选的速度参数。聚焦位置和速度默认为一个无单位的类型。或者，如果支持的话，定位可能会被要求在 m-1 个单位。

Relative –规定距离参数和可选的速度参数。负距离参数意味着反方向。
Continuous –规定速度参数。负速度参数意味着反方向。

表 114：Move (focus)命令

Move	请求-响应
消息名称	描述
MoveRequest	VideoSource 移动（焦点）操作要求的参考。 tt:ReferenceToken VideoSourceToken[1][1] tt:FocusMove Focus [1][1]
MoveResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoSource	请求的视频源不存在。
env:Receiver ter:ActionNotSupported ter:NoImagingForSource	请求的视频源不支持图像设置。

10.1.5 获取运行选项

GetMoveOptions 命令可用于检索在第 10.1.4 节定义的 Move 命令对对焦镜头的移动选项。支持镜头移动操作的设备也应支持 GetMoveOptions 命令。

表 115：GetMoveOptions (focus)命令

GetMoveOptions	请求-响应
消息名称	描述
GetMoveOptions	视频源对请求移动选项的参考。 tt:ReferenceToken VideoSourceToken[1][1]
GetMoveOptionsResponse	此消息包含对焦镜头移动选项的有效范围。 tt:MoveOptions20 MoveOptions[1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoSource	请求的视频源不存在。
env:Receiver ter:ActionNotSupported ter:NoImagingForSource	请求的视频源不支持图像设置。

10.1.6 停止

Stop 命令停止所有正在进行的透镜焦点运动。如果设备支持对焦，则可以通过停止操作停止对焦。操作不会影响正在进行的自动对焦操作。

表 116：Stop (focus)命令

Stop	请求-响应
消息名称	描述
StopRequest	对焦运动在哪停止的视频源的参考。 tt:ReferenceToken VideoSourceToken[1][1]
StopResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoSource	请求的视频源不存在。
env:Receiver ter:ActionNotSupported ter:NoImagingForSource	请求的视频源不支持图像设置。

10.1.7 获取图像状态

GetStatus 命令请求当前的图像设备状态。如果设备支持焦点移动控制，那么它可以通过 GetStatus 命令得到有效的图像状态。

图像状态包含：

- 焦点位置，移动状态和错误信息。
  - 表示无单位类型的焦点位置。
  - 移动状态可能是在运行中，空闲或未知的状态。
  - 通过一个字符串提供错误信息，例如：a positioning error indicated by the hardware。

表 117：GetStatus (focus)命令

GetStatus	请求-响应
消息名称	描述
GetStatusRequest	此消息包含要求成像状态的视频源的参考。 tt:VideoSourceToken VideoSourceToken[1][1]
GetStatusResponse	此消息包含所要求的成像状态。 tt:ImagingStatus20 ImagingStatus [1][1]

错误码	描述
env:Sender ter:InvalidArgVal ter:NoSource	请求的视频源不存在。
env:Receiver ter:ActionNotSupported ter:NoImagingForSource	请求的视频源不支持图像设置。

10.2 服务错误码

表 118 列出了影像服务的具体的错误代码。此外，每个命令还可以生成一个通用的错误，见表 6。

具体错误是作为一个通用的错误子码定义的，请参阅第 5.11.2.1 。母公司通用的子码是下面每一行的顶部的子码，具体的错误子码是在最底层。

表 118：成像的具体错误码

错误码	母码 子码	错误原因	描述
env:Receiver	ter:ActionNotSupported ter:NoImagingForSource	VideoSource 不支持成像设置。	请求的 VideoSource 不支持成像设置。
env:Sender	ter:InvalidArgVal ter:SettingsInvalid	无效的配置	请求的设置不正确。
env:Sender	ter:InvalidArgVal ter:NoSource	视频源不存在。	请求的视频源不存在。

11 媒体配置

媒体服务用于配置 NVT 的流媒体属性。NVT 支持[ONVIF Media WSDL]指定的媒体服务。媒体服务允许客户端配置媒体和其他实时流配置。媒体配置通过媒体文件处理。 ONVIF 的媒体配置模型请参阅 4.8 节。

媒体服务指令分为两大类：

- 媒体配置：
  - Media profile commands（媒体文件命令）
  - Video source commands（视频源命令）
  - Vide encoder commands（视频编码器命令）
  - Audio source commands（音频源命令）
  - Audio encoder commands（音频编码器命令）
  - Video analytics commands（视频分析命令）
  - Metadata commands（元数据命令）
  - Audio output commands（音频输出命令）
  - Audio decoder commands（音频解码器命令）

- 流媒体:
  - Request stream URI （请求 URI 流）
  - Get snapshot URI （获取 URI 快照）
  - Multicast control commands （组播控制命令）
  - Media synchronization point（媒体同步）

11.1 音视频编解码器

NVT 流的音频和视频数据使用适当的编码算法。NVT 也能解码音频。NVT 根据制造商的选择支持任何音频和视频编解码器，比特率和分辨率。为了确保客户端和 NVT 之间的互操作性，标准规定以下的编解码器配置文件：

- NVT 支持 JPEG QVGA。
- NVT 支持 G.711 μ Law（仅简单摄像机话筒，1ch）[ITU-T G.711]

11.2 媒体文件

媒体文件包含了一套媒体配置。媒体文件被 NVT 的客户端媒体流配置属性所使用。

NVT 在启动时提供至少有一个媒体文件。 NVT 上应提供可以即用的，包含设备最常见的媒体配置的配置文件。

文件由一组相互关联的配置实体组成。配置由 NVT 提供，可以由 NVT 的静态或动态创建。例如，动态的配置可由 NVT 根据当前有效的编码器资源创建。配置实体为以下之一：

- Video source configuration（视频源配置）
- Audio source configuration（音频源配置）
- Video encoder configuration（视频编码器配置）
- Audio encoder configuration（音频编码器配置）
- PTZ configuration（PTZ 配置）
- Video analytics configuration（视频分析配置）
- Metadata configuration（流媒体配置）
- Audio output configuration（音频输出配置）
- Audio decoder configuration（音频解码器配置）

文件包括所有这些配置实体或者一个子集。根据 NVT 的性能，一个特定的配置实体可以是文件的一部分。例如，音频源和音频编码器配置的文件只能存在于支持音频的设备。

11.2.1 创建媒体文件

此操作将创建一个新的空白媒体文件。媒体文件建立在 NVT 中且持续的（在重新启动后保持）。NVT 支持通过 CreateProfile 命令创建在本标准定义的媒体文件。

创建的文件是可删除的，NVT 可以对返回的文件的“fixed”属性设置成无效。

表 119：CreateProfile 命令

CreateProfile	请求-响应
消息名称	描述
CreateProfileRequest	包含友好的要创建的文件的名称以及一个可选的令牌参数，指定唯一标识符的新的媒体文件。 tt:Name Name [1][1] tt:ReferenceToken Token [0][1]
CreateProfileResponse	没有配置实体返回一个空的文件结构。 tt:Profile Profile [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:ProfileExists	文件令牌 ProfileToken 已经存在。
env:Receiver ter:Action ter:MaxNVTProfiles	支持的文件已达到最大数量。

11.2.2 获取多个媒体文件

任何端点都可以使用 GetProfiles 命令请求 NVT 现有的媒体文件。预配置或动态配置文件都可以使用此命令检索。该命令列出设备所有配置的文件。客户端不需要知道媒体文件就能使用该命令。NVT 支持通过 GetProfiles 命令检索媒体文件。

NVT 在所有传回的资料元素中包括“fixed”属性。

表 120：GetProfiles 命令

GetProfiles	请求-响应
消息名称	描述
GetProfilesRequest	空消息。
GetProfilesResponse	响应包含一个文件清单。每个文件包含了一套可用于流媒体、分析学、元数据流等定义了特定配置的配置实体。 tt:Profile Profiles [0][unbounded]
错误码	描述
	没有具体的错误命令！

11.2.3 获取媒体文件

如果文件令牌是已知的，则可以通过 GetProfile 命令获取文件。NVT 支持通过 GetProfile 命令检索特定的媒体文件。

NVT 在传回的资料元素中包括“fixed”属性。

表 121：GetProfile 命令

GetProfiles	请求-响应
消息名称	描述
GetProfilesRequest	此消息包含所要求的文件令牌。 tt:ReferenceToken ProfileToken [1][1]
GetProfilesResponse	响应中包含表明令牌参数的文件。每个文件包含了一套可用于流媒体，分析学，元数据流等定义了特定配置的配置实体。 tt:Profile Profile [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。

11.2.4 添加视频源配置

此操作增加了一个视频源配置到现有的媒体文件。如果这样的配置存在于媒体文件，它会被永久替换。NVT 支持通过 AddVideoSourceConfiguration 命令添加文件的视频源配置。

表 122：AddVideoSourceConfiguration 命令

AddVideoSourceConfiguration	请求-响应
消息名称	描述
AddVideoSourceConfigurationRequest	包含新添加的视频源配置的参考和目标文件。 tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]
AddVideoSourceConfigurationResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	配置令牌表示的视频源配置不存在。
env:Receiver ter:Action ter:ConfigurationConflict	其他媒体文件的配置和新添加的冲突，继续添加会引起媒体文件冲突。

11.2.5 添加视频编码器配置

此操作添加了视频编码器配置到现有媒体文件。如果配置存在于媒体文件，它会被永久替换。NVT 支持通过 AddVideoEncoderConfiguration 命令添加视频编码器配置到文件。

添加视频编码器配置到文件是指一个使用配置文件的流将包含该视频数据。视频编码器配置应在添加一个视频源配置后被添加。

表 123：AddVideoEncoderConfiguration 命令

AddVideoEncoderConfiguration	请求-响应
消息名称	描述
AddVideoEncoderConfigurationRequest	包含新添加的视频编码器配置的参考和目标文件。 tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]
AddVideoEncoderConfigurationResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	配置令牌表示的视频编码器配置不存在。
env:Receiver ter:Action ter:ConfigurationConflict	其他媒体文件的配置和新添加一个冲突，继续添加会引起媒体文件冲突。

11.2.6 添加音频源配置

此操作将添加音频源配置到一个现有的媒体文件。如果配置在媒体文件存在，它会被永久替换。支持从 NVT 到客户的音频流 NVT 支持通过 AddAudioSourceConfiguration 命令添加音频源配置到文件。

表 124：AddAudioSourceConfiguration 命令

AddAudioSourceConfiguration	请求-响应
消息名称	描述
AddAudioSourceConfigurationRequest	包含新添加的视频编码器配置的参考和目标文件。 tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]
AddAudioSourceConfigurationResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	ConfigurationToken 表示的音频源配置不存在
env:Receiver ter:Action ter:ConfigurationConflict	其他媒体文件的配置和新添加的冲突，继续添加会引起媒体文件冲突。
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	不支持音频。

11.2.7 添加音频源编码器配置

此操作将添加音频编码器配置到一个现有的媒体文件。如果配置存在于媒体文件，它会被永久替换。支持从 NVT 到客户端的音频流的 NVT 支持通过 AddAudioEncoderConfiguration 命令添加音频编码器配置到配置文件。

将音频编码器配置添加到媒体文件意味着使用媒体文件的流将包含该音频数据。音频编码器配置在添加一个音频源配置后被添加。

表 125：AddAudioEncoderConfiguration 命令

AddAudioEncoderConfiguration	请求-响应
消息名称	描述
AddAudioEncoderConfigurationRequest	包含新添加的音频编码器配置的参考和目标文件。 tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]
AddAudioEncoderConfigurationResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	配置令牌表示的音频编码配置不存在。
env:Receiver ter:Action ter:ConfigurationConflict	其他媒体文件的配置和新添加的冲突，继续添加会引起媒体文件冲突。
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	不支持音频。

11.2.8 添加云台配置

此操作增加了 PTZ 配置到现有的媒体文件。如果配置在媒体文件上存在，它会被永久替换。支持 PTZ 控制的 NVT 通过 AddPTZConfiguration 命令支持 PTZ 添加配置文件。

添加 PTZ 配置到媒体文件意味着使用该媒体文件的流可以包含 PTZ 状态（元数据），以及媒体文件可用于控制 PTZ 运动，见第 16 章。

表 126：AddPTZConfiguration 命令

AddPTZConfiguration	请求-响应
消息名称	描述
AddPTZConfigurationRequest	包含新添加的云台配置的参考和目标文件。 tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]
AddPTZConfigurationResponse	空消息。
错误码	描述

env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	配置令牌表示的 PTZ 配置不存在。
env:Receiver ter:Action ter:ConfigurationConflict	其他媒体文件的配置和新添加的冲突，继续添加会引起媒体文件冲突。
env:Receiver ter:ActionNotSupported ter:PTZNotSupported	不支持 PTZ。

11.2.9 添加视频分析配置

此操作添加了视频分析配置到现有的媒体文件。如果配置在媒体文件存在，它会被永久替换。支持视频分析的 NVT 支持通过 AddVideoAnalyticsConfiguration 命令添加视频分析配置到配置文件。

添加视频分析配置到媒体文件意味着使用该媒体配置文件的流可以包含视频分析数据（元数据）所提交的参考配置定义。视频分析数据指定在第 17.1 节，分析配置是通过在 11.9 节中定义的命令管理的。

文件只包含一个视频分析配置，而没有视频信号源配置是不完整的。因此，客户应先添加一个视频源配置一个文件，然后再添加视频分析配置。添加视频源配置之前 NVT 拒绝添加视频分析配置。在这种情况下，它会提示与配置冲突的错误。

表 127：AddVideoAnalytics 命令

AddVideoAnalytics	请求-响应
消息名称	描述
AddVideoAnalyticsRequest	包含新添加的视频分析的参考和目标文件。 tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]
AddVideoAnalyticsResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	配置令牌表示的 VideoAnalytics 不存在。
env:Receiver ter:Action ter:ConfigurationConflict	其他媒体文件的配置和新添加的冲突，继续添加会引起媒体文件冲突。
env:Receiver ter:ActionNotSupported ter: VideoAnalyticsNotSupported	不支持视频分析。

11.2.10 添加元数据配置

此操作添加了元数据配置到现有的媒体文件。如果配置在媒体文件上存在，它会永久代替。NVT 支持文件通过 AddMetadataConfiguration 命令添加一个元数据配置。

添加元数据配置到文件中意味着使用该文件流时包含元数据。元数据可以包含事件，PTZ 状态，和/或视频分析数据。处理元数据配置通过在 11.10 和 11.9.4 节定义的命令。

表 128：AddMetadataConfiguration 命令

AddMetadataConfiguration	请求-响应
消息名称	描述
AddMetadataConfiguration Request	包含新添加的元数据配置参考和目标文件。 tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]
AddMetadataConfigurationResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	配置令牌表示的数据流配置不存在。

env:Receiver ter:Action ter:ConfigurationConflict	其他媒体文件的配置和新添加的冲突，继续添加会引起媒体文件冲突。

11.2.11 添加音频输出配置

此操作添加了一个音频输出配置到现有的媒体文件。如果配置在媒体文件存在，它会被永久替换。一个有音频输出的 NVT 支持通过 AddAudioOutputConfiguration 命令添加音频输出配置到文件。

表 129： AddAudioOutputConfiguration 命令

AddAudioOutputConfiguration	请求-响应
消息名称	描述
AddAudioOutputConfigurationRequest	包含添加音频输出配置参考和目标文件。 tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]
AddAudioOutputConfigurationResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	配置令牌表示的音频输出配置不存在。
env:Receiver ter:Action ter:ConfigurationConflict	其他媒体文件的配置和新添加的冲突，继续添加会引起媒体文件冲突。
env:Receiver ter:ActionNotSupported ter: AudioOutputNotSupported	不支持音频或音频输出。

11.2.12 添加音频解码器配置

此操作添加音频解码器配置到现有的媒体文件。如果有一个配置在媒体文件，它将永久替换。有音频解码性能的 NVT 支持通过 AddAudioDecoderConfiguration 命令添加音频解码器配置给文件。

表 130： AddAudioDecoderConfiguration 命令

AddAudioDecoderConfiguration	请求-响应
消息名称	描述
AddAudioDecoderConfigurationRequest	包含添加音频解码器配置参考和目标文件。 tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]
AddAudioDecoderConfigurationRespons e	空消息。
错误码	描述

env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	配置令牌表示的音频解码器配置不存在。
env:Receiver ter:Action ter:ConfigurationConflict	其他媒体文件的配置和新添加的冲突，继续添加会引起媒体文件冲突。
env:Receiver ter:ActionNotSupported ter: AudioDecodingNotSupported	不支持音频或音频解码。

11.2.13 移除视频源配置

此操作会从现有的媒体文件的删除一个视频源配置。如果媒体文件不包含一个视频源配置，操作无效。删除是永久性的。NVT 支持通过 RemoveVideoSourceConfiguration 命令从文件删除视频源配置。

视频源配置仅在视频编码器从媒体文件删除后才能被删除。

表 131： RemoveVideoSourceConfiguration 命令

RemoveVideoSourceConfiguration	请求-响应
消息名称	描述
RemoveVideoSourceConfigurationRequest	包含应被删除的视频源配置的媒体文件参考。 tt:ReferenceToken ProfileToken [1][1]
RemoveVideoSourceConfigurationResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	当前的媒体文件不存在视频源配置。
env:Receiver ter:Action ter:ConfigurationConflict	其他媒体文件的配置依赖于视频源配置，删除它会导致媒体文件的冲突。

11.2.14 移除视频源编码器配置

此操作会从一个现有的媒体文件删除视频编码器配置。如果媒体文件不包含视频编码器的配置，操作无效。删除具有永久性。NVT 支持通过 RemoveVideoEncoderConfiguration 命令从文件删除视频编码器配置。

表 132：RemoveVideoEncoderConfiguration 命令

RemoveVideoEncoderConfiguration	请求-响应
消息名称	描述

RemoveVideoEncoderConfigurationRequest	包含应被删除的视频解码器配置的媒体文件参考。 tt:ReferenceToken ProfileToken [1][1]
RemoveVideoEncoderConfigurationResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	当前的媒体文件不存在视频解码器配置。
env:Receiver ter:Action ter:ConfigurationConflict	其他媒体文件的配置依赖于视频解码器配置，删除它会导致媒体文件的冲突。

11.2.15 移除音频源编码器配置

此操作会从一个现有的媒体文件删除音频源配置。如果媒体文件不包含音频信号源的配置，操作无效。删除是永久性的。支持音频流从 NVT 到客户端的 NVT 支持通过 RemoveAudioSourceConfiguration 命令从文件删除音频源配置。

仅支持从媒体文件删除音频编码器配置后删除音频源配置。

表 133：RemoveAudioSourceConfiguration 命令

RemoveAudioSourceConfiguration	请求-响应
消息名称	描述
RemoveAudioSourceConfigurationRequest	包含应被删除的音频源配置的媒体文件参考。 tt:ReferenceToken ProfileToken [1][1]
RemoveAudioSourceConfigurationResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	当前的媒体文件不存在音频源配置。
env:Receiver ter:Action ter:ConfigurationConflict	其他媒体文件的配置依赖于音频源配置，删除它会导致媒体文件的冲突。
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	不支持音频。

11.2.16 移除音频编码器配置

此操作会从一个现有的媒体文件删除音频编码器配置。如果媒体文件不包含音频编码器的配置，操作无效。删除具有永久性。NVT 支持通过 RemoveAudioEncoderConfiguration 命令

从文件删除音频编码器配置。

表 134：RemoveAudioEncoderConfiguration 命令

RemoveAudioEncoderConfiguration	请求-响应
消息名称	描述
RemoveAudioEncoderConfigurationRequest	包含应被删除的音频解码器配置的媒体文件参考。 tt:ReferenceToken ProfileToken [1][1]
RemoveAudioEncoderConfigurationResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	当前的媒体文件不存在音频解码器配置。
env:Receiver ter:Action ter:ConfigurationConflict	其他媒体文件的配置依赖于音频解码器配置，删除它会导致媒体文件的冲突。
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	不支持音频。

11.2.17 移除云台配置

此操作会从现有的媒体文件删除 PTZ 配置。如果媒体文件不包含一个 PTZ 配置，操作无效。删除是永久性的。支持 PTZ 控制的 NVT 支持通过 RemovePTZConfiguration 命令从文件删除 PTZ 配置。

表 135：RemovePTZConfiguration 命令

RemovePTZConfiguration	请求-响应
消息名称	描述
RemovePTZConfigurationRequest	包含应被删除的 PTZ 配置的媒体文件参考。 tt:ReferenceToken ProfileToken [1][1]
RemovePTZConfigurationResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	当前的媒体文件不存在 PTZ 配置。
env:Receiver ter:Action	其他媒体文件的配置依赖于 PTZ 配置，删除它会导致媒体文件的冲突。

ter:ConfigurationConflict	
env:Receiver ter:ActionNotSupported ter: PTZNotSupported	不支持 PTZ。

11.2.18 移除视频分析配置

此操作会从现有的媒体文件删除视频分析配置。如果媒体文件不包含一个视频分析配置，操作无效。删除是永久性的。支持视频分析的 NVT 支持通过 RemoveVideoAnalyticsConfiguration 命令从文件删除视频分析配置。

表 136：RemoveVideoAnalyticsConfiguration 命令

RemoveVideoAnalyticsConfiguration	请求-响应
消息名称	描述
RemoveVideoAnalyticsConfigurationRequest	包含应被删除的视频分析配置的媒体文件参考。 tt:ReferenceToken ProfileToken [1][1]
RemoveVideoAnalyticsConfigurationResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	当前的媒体文件不存在视频分析配置。
env:Receiver ter:Action ter:ConfigurationConflict	其他媒体文件的配置依赖于视频分析配置，删除它会导致媒体文件的冲突。
env:Receiver ter:ActionNotSupported ter:VideoAnalyticsNotSupported	不支持视频分析。

11.2.19 移除元数据配置

此操作会从现有的媒体文件删除元数据配置。如果媒体文件不包含一个元数据配置，操作无效。删除是永久性的。NVT 支持通过 RemoveMetadataConfiguration 命令从文件删除元数据配置。

表 137：RemoveMetadataConfiguration 命令

RemoveMetadataConfiguration	请求-响应
消息名称	描述
RemoveMetadataConfigurationRequest	包含应被删除的元数据配置的媒体文件参考。 tt:ReferenceToken ProfileToken [1][1]
RemoveMetadataConfigurationResponse	空消息。

错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	当前的媒体文件不存在元数据配置。
env:Receiver ter:Action ter:ConfigurationConflict	其他媒体文件的配置依赖于元数据配置，删除它会导致媒体文件的冲突。

11.2.20 移除音频输出配置

此操作会从现有的媒体文件删除音频输出配置。如果媒体文件不包含一个音频输出配置，操作无效。删除是永久性的。具有至少一个音频输出的 NVT 支持通过 RemoveAudioOutputConfiguration 命令从文件删除音频输出配置。

表 138：RemoveAudioOutputConfiguration 命令

RemoveAudioOutputConfiguration	请求-响应
消息名称	描述
RemoveAudioOutputConfigurationRequest	包含应被删除的音频输出配置的媒体文件参考。 tt:ReferenceToken ProfileToken [1][1]
RemoveAudioOutputConfigurationResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	当前的媒体文件不存在音频输出配置。
env:Receiver ter:Action ter:ConfigurationConflict	其他媒体文件的配置依赖于音频输出配置，删除它会导致媒体文件的冲突。
env:Receiver ter:ActionNotSupported ter: AudioOutputNotSupported	不支持音频或音频输出。

11.2.21 移除音频编码器配置

此操作会从现有的媒体文件删除音频解码器配置。如果媒体文件不包含一个音频解码器配置，操作无效。删除是永久性的。支持音频解码的 NVT 支持通过 RemoveAudioDecoderConfiguration 命令从文件删除音频解码器配置。

表 139：RemoveAudioDecoderConfiguration 命令

RemoveAudioDecoderConfiguration	请求-响应
消息名称	描述
RemoveAudioDecoderConfigurationRequest	包含应被删除的音频解码器配置的媒体文件参考。 tt:ReferenceToken ProfileToken [1][1]
RemoveAudioDecoderConfigurationResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	当前的媒体文件不存在音频解码器配置。
env:Receiver ter:Action ter:ConfigurationConflict	其他媒体文件的配置依赖于音频解码器配置，删除它会导致媒体文件的冲突。
env:Receiver ter:ActionNotSupported ter: AudioDecodingNotSupported	不支持音频或音频解码。

11.2.22 删除媒体文件

此操作永久删除文件。NVT 支持通过 DeleteProfile 命令删除媒体文件。

表 140：DeleteProfile 命令

DeleteProfile	请求-响应
消息名称	描述
DeleteProfileRequest	包含应被删除的媒体文件的 ProfileToken。 tt:ReferenceToken ProfileToken [1][1]
DeleteProfileResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	所请求的文件令牌 ProfileToken 不存在。
env:Sender ter:Action ter:DeletionOfFixedProfile	固定文件不能被删除。

11.3 视频源

代表未编码的视频源的视频输入。结构中包含视频的像素分辨率、帧率和图像设置。例如，例如，如果图像设置支持和包含对焦、曝光和亮度参数，图像设置可以通过图像服务实现。详情请参阅第 10 章。

11.3.1 获取视频源集

此操作列出了所有有效的视频信号源设备。NVT 支持通过 GetVideoSources 命令列出有效视频源。

表 141：GetVideoSources 命令

GetVideoSources	请求-响应
消息名称	描述
GetVideoSourcesRequest	空消息。
GetVideoSourcesResponse	包含了所有有效的视频源配置描述结构清单。 tt:VideoSource VideoSources [0][unbounded]
错误码	描述
	没有具体的错误命令。

11.4 视频源配置

一个视频源配置包含视频源参考和边界结构，其中边界结构是指视频源像素全部或者一部分面积。该配置定义传输给客户的视频流。如果视频源配置在配置文件中使用，它的 UseCount 需要加 1，表示改变该配置可能影响其它用户。

11.4.1 获取视频源配置集

此操作列出了设备现有的 NVT 的所有视频源配置。客户端不需要知道任何有关的视频源配置，就可以使用该命令。NVT 支持通过 GetVideoSourceConfigurations 命令列出有效视频源配置。

表 142：GetVideoSourceConfigurations 命令

GetVideoSourcesConfigurations	请求-响应
消息名称	描述
GetVideoSourcesConfigurations-Request	空消息。
GetVideoSourcesConfigurations-Response	此消息包含 NVT 所有有效的视频源配置描述结构清单。视频源配置不总是指示 SourceToken 元素的实时视频源。 tt:VideoSourceConfigurations Configurations[0][unbounded]
错误码	描述
	没有具体的错误命令。

11.4.2 获取视频源配置

如果视频源配置令牌是已知的，NVT 支持通过 GetVideoSourceConfiguration 命令检索获取视频信号源的配置。

表 143：GetVideoSourceConfiguration 命令

GetVideoSourcesConfiguration	请求-响应
消息名称	描述
GetVideoSourcesConfigurationRequest	此消息包含要求的视频源配置的令牌。 tt:ReferenceToken ConfigurationToken [1][1]

GetVideoSourcesConfigurationResponse	此消息包含与请求的令牌匹配的视频源配置。视频源配置不总是指示源令牌元素的实时视频源。 tt:VideoSourceConfiguration Configurations [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	表明请求配置的配置令牌不存在。

11.4.3 获取多个兼容视频源配置

此操作请求获取 NVT 所有的，与对应的媒体文件兼容的视频源配置。每个返回的配置应是一个有效的可以作为媒体文件上的 AddVideoSourceConfiguration 命令的输入参数。结果因设备的配置和设置情况而不同。NVT 支持通过 GetCompatibleVideoSourceConfigurations 命令列出（与特定文件）兼容的视频源配置。

表 144：GetCompatibleVideoSourceConfigurations 命令

GetCompatibleVideoSource-Configurations	请求-响应
消息名称	描述
GetCompatibleVideoSource-ConfigurationsRequest	包含现有媒体文件的令牌。 tt:ReferenceToken ProfileToken [1][1]
GetCompatibleVideoSource-ConfigurationsResponse	包含一个兼容媒体文件的视频源配置清单。 tt:VideoSourceConfiguration Configurations [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	表明请求的文件令牌 ProfileToken 不存在。

11.4.4 获取视频源配置选项

当视频源参数被重新配置是，该操作返回有效的配置选项，如果使用了特定的视频源配置，选项应该关注这些特定的配置。指定的媒体文件选项应与该媒体文件兼容。NVT 支持通过 GetVideoSourceConfigurationOptions 命令列出有效视频源参数选项（对于一个给定的文件和配置）。

表 145：GetVideoSourceConfigurationOptions 命令

GetVideoSourceConfigurationOptions	请求-响应
消息名称	描述
GetVideoSourceConfiguration-OptionsRequest	此消息包含一个视频源配置的可选令牌和媒体文件。 ConfigurationToken 指定一个现有的配置选项。 ProfileToken 所指定的选项应当是兼容现有的媒体文件。 tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]
GetVideoSourceConfiguration-OptionsResponse	此消息包含视频配置选项。视频源配置选项指定特定的配置。指定的媒体文件选项应与该媒体文件兼容。如果没有指定标记，选项应被视为通用设备。

	tt:VideoSourceConfigurationOptions Options [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	表明请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	请求配置不存在。

11.4.5 设置视频源配置

此操作修改一个视频源配置。ForcePersistence 标志表示 NVT 重启后是否保留这些更改。使用该配置运行的流可能立即更新配置。除非客户端请求一个新的 URI 流，并重新启动任何受影响的流，否则所做的更改不保证生效。改变运行流的 NVC 方法超出本规范的范围。NVT 支持通过 SetVideoSourceConfiguration 命令修改视频源参数。

表 146：SetVideoSourceConfiguration 命令

SetVideoSourceConfiguration	请求-响应
消息名称	描述
SetVideoSourceConfiguration Request	此消息包含一个视频源配置的可选令牌和媒体文件。 ConfigurationToken 指定一个现有的配置选项。 ProfileToken 所指定的选项应当是兼容现有的媒体文件。 tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]
SetVideoSourceConfigurationResponse	此消息包含视频配置选项。视频源配置选项指定特定的配置。指定的媒体文件选项应与该媒体文件兼容。如果没有指定标记，选项应被视为通用设备。 tt:VideoSourceConfigurationOptions Options [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	配置不存在。
env:Sender ter:InvalidArgVal ter:ConfigModify	配置参数无法设置。
env:Receiver ter:Action ter:ConfigurationConflict	新设置与其他配置冲突。

11.5 视频编码器配置

一个 VideoEncoderConfiguration 包含以下视频编码数据参数配置：

- Encoder –用于视频数据的编码器。
- Resolution –用于编码的视频数据像素分辨率。

- Quality –决定视频的质量。的支持质量范围的值越高意味着更高的质量。
- RateControl –配置比特率[kbps]的定义参数，以及一个编码间隔参数（间隔图像进行编码和传输）和 FrameRateLimit[FPS]参数来配置输出帧率。
- MPEG4/H264 specifics –定义编码文件和 GOV 长度[帧] 。

视频编码器配置结构还包含组播参数和会话超时来定义视频流的行为。如果视频编码器配置在配置文件中使用，它的 UseCount 需要加 1，表示改变该配置可能影响其它用户。

11.5.1 获取多个视频编码器配置

此操作列出 NVT 所有现有的视频编码器配置。该命令列出设备的所有配置的视频编码器配置。客户端不需要事先知道任何有关的视频编码器配置就能使用该命令。NVT 支持通过 GetVideoEncoderConfigurations 命令列出现有的视频编码器配置。

表 147：GetVideoEncoderConfigurations 命令

GetVideoEncoderConfigurations	请求-响应
消息名称	描述
GetVideoEncoderConfigurations-Request	空消息。
GetVideoEncoderConfigurations-Response	此消息包含 NVT 所有现有的视频编码器清单配置。 tt:VideoEncoderConfiguration Configurations [0][unbounded]
错误码	描述
	没有具体的错误命令。

11.5.2 获取视频编码器配置

如果视频编码器配置令牌是已知的，可以通过 GetVideoEncoderConfiguration 命令获取编码器的配置。NVT 支持通过 GetVideoEncoderConfiguration 命令检索一个特定的视频编码器配置。

表 148：GetVideoEncoderConfiguration 命令

GetVideoEncoderConfiguration	请求-响应
消息名称	描述
GetVideoEncoderConfigurationRequest	此消息包含要求的视频编码器配置令牌。 tt:ReferenceToken ConfigurationToken [1][1]
GetVideoEncoderConfigurationResponse	此消息包含请求视频编码器配置的匹配令牌。 tt:VideoEncoderConfiguration Configuration [1][1]
错误码	描述
env:Sender ter:InvalidArgVa ter:NoConfig	表示请求的配置的 ConfigurationToken 不存在。

11.5.3 获取多个兼容视频解码器配置

此操作列出 NVT 所有与媒体文件兼容的视频编码器配置。每个返回的配置可以作为一

个在媒体文件 AddVideoEncoderConfiguration 命令有效的输入参数。返回结果因设备的功能，配置和设置不同而有变化。NVT 支持通过 GetCompatibleVideoEncoderConfigurations 命令列出（与特定的配置文件）兼容的视频编码器配置。

表 149：GetCompatibleVideoEncoderConfigurations 命令

GetCompatibleVideoEncoderConfigurations	请求-响应
消息名称	描述
GetCompatibleVideoEncoder-ConfigurationsRequest	包含要求的媒体文件令牌。 tt:ReferenceToken ProfileToken [1][1]
GetCompatibleVideoEncoder-ConfigurationsResponse	包含一个兼容媒体文件的视频编码器配置清单。 tt:VideoEncoderConfiguration Configurations[0][unbounded]
错误码	描述
env:Sender ter:InvalidArgVa ter:NoProfile	表示请求的文件令牌 ProfileToken 不存在。

11.5.4 获取视频编码器配置选项集

视频编码器参数的重新配置时，该操作返回有效的选项。NVT 支持通过 GetVideoEncoderConfigurationOptions 命令列出视频参数选项(对于一个给定的文件和配置)。

表 150：GetVideoEncoderConfigurationOptions 命令

GetVideoEncoderConfigurationOptions	请求-响应
消息名称	描述
GetVideoEncoderConfigurationOptions-Request	此消息包含一个可选的视频编码器配置令牌和媒体文件。 ConfigurationToken 指定一个现有的配置选项。 ProfileToken 指定一个现有的兼容媒体文件的选项。 tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]

GetVideoEncoderConfigurationOptions-Response	此消息包含视频配置选项。选项指定视频编码器配置的特定配置。媒体文件指定的选项应与媒体文件兼容。如果没有指定标记，选项被视为通用设备。 tt:VideoEncoderConfi gurationOptions Options [1][1]
错误码	描述
env:Sender ter:InvalidArgVa ter:NoProfile	表示请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	请求的配置不存在。

11.5.5 修改视频编码器配置

此操作修改视频编码配置。ForcePersistence 标志表示重新启动 NVT 后是否保存更改。组播设置的改变是永久的。使用此配置的运行流可立即根据新的设置更新，但不能保证所做的更改生效，除非客户端请求一个新的 URI 流并重新启动任何受影响的流。如果新设置使，用 RTSP 谈判达成的任何参数无效，例如改变编解码器类型的任何参数，NVT 都不应用这些设置到现有的流。相反，它必须使用旧的设置继续运行流或对受影响的流停止发送数据。

用 NVC 改变运行流的方法超出本规范的范围。NVT 支持通过 SetVideoEncoderConfiguration 命令修改视频编码参数。

表 151：SetVideoEncoderConfiguration 命令

SetVideoEncoderConfiguration	请求-响应
消息名称	描述
SetVideoEncoderConfiguration-Request	配置元素包含修改后的视频编码器配置。配置在 NVT 存在。ForcePersistence 元素决定是否在重启后存储和保持配置更改。如果为真，改变是永久的。如果为假，可能会在重新启动后恢复到改变以前的值。 tt:VideoEncoderCo nfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]
SetVideoEncoderConfiguration-Response	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	请求的配置不存在。
env:Sender ter:InvalidArgVal ter:ConfigModify	请求的配置不能设置。
env:Receiver ter:Action	新设置与其他使用的设置冲突。

ter:ConfigurationConflict	

11.5.6 获取有效的视频编码数量

GetGuaranteedNumberOfVideoEncoderInstances 命令可用于请求每个视频源配置（应用程序）保证视频编码器实例的最低数量。NVT 支持此命令。ONVIF 1.02 中添加了此命令。

表 152：GetGuaranteedNumberO fVideoEncoderInstances 命令

GetGuaranteedNumberO fVideoEncoderInstances	请求-响应
消息名称	描述
GetGuaranteedNumberOf-EncoderInstancesRequest	这个请求包含一个视频源配置的令牌。 tt: ReferenceToken ConfigurationToken [1][1]
GetGuaranteedNumberOf-EncoderInstancesResponse	此消息中包含每个视频源配置（应用程序）的最低保证总数的编码实例。设备限制了各自的视频编解码器的实例数量，响应包含多少 JPEG，H264 和 MPEG4 可在同一时间设置。在其他所有情况下，该设备能够在同一时间独立的从配置的视频编解码器提供流的总数。 xs:int TotalNumber [1][1] xs:int JPEG [0][1] xs:int H264 [0][1] xs:int MPEG4 [0][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	请求的配置配置令牌不存在。

11.6 音频源

音频源代表未编码的音频输入和表明输入通道的数量。

11.6.1 获取多个音频源

此操作会列出设备所有有效的音频源。支持音频流从 NVT 到客户端的 NVT 设备支持通过 GetAudioSources 命令列出有效的音频源。

表 153：GetAudioSources 命令

GetAudioSources	请求-响应
消息名称	描述
GetAudioSourcesRequest	空消息
GetAudioSourcesResponse	包含一个描述所有有效的音频源设备的结构清单。 tt:AudioSource AudioSources[0][unbounded]
错误码	描述

env:Receiver ter:ActionNotSupported ter:AudioNotSupported	NVT 不支持音频。

11.7 音频源配置

音频源配置包含了输入媒体文件的音频源参考。如果音频源配置在配置文件中使用，它的 UseCount 需要加 1，表示改变该配置可能影响其它用户。

11.7.1 获取多个音频源配置

此操作列出 NVT 设备所有现有的音频源配置。客户端不需要知道任何有关音频源的配置就能使用该命令。支持音频流从 NVT 到客户端的 NVT 设备，支持通过 GetAudioSourceConfigurations 命令列出有效的音频源配置。

表 154：GetAudioSourceConfigurations 命令

GetAudioSourceConfigurations	请求-响应
消息名称	描述
GetAudioSourceConfigurations-Request	空消息。
GetAudioSourceConfigurations-Response	此消息包含了 NVT 所有现有的音频源配置清单。音频源配置 SourceToken 元素总是指着一个音频源。 tt:AudioSourceConfiguration Configurations [0][unbounded]
错误码	描述
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	NVT 不支持音频。

11.7.2 获取音频源配置

GetAudioSourceConfiguration 命令获取音频源配置令牌已知的音频源配置。支持音频流从 NVT 到客户端的 NVT 支持通过 GetAudioSourceConfiguration 命令检索指定的音频源配置。

表 155：GetAudioSourceConfiguration 命令

GetAudioSourceConfiguration	请求-响应
消息名称	描述
GetAudioSourceConfiguration-Request	此消息包含所要求的音频源配置令牌。音频源配置的源令牌元素总是指着一个音频源。 tt:ReferenceToken ConfigurationToken [1][1]
GetAudioSourceConfiguration-Response	此消息包含了 NVT 所有现有的音频源配置清单。音频源配置的源令牌元素总是指在一个音频源。 tt:AudioSourceConfiguration Configurations [0][unbounded]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	表明请求的配置的配置令牌不存在的。

env:Receiver ter:ActionNotSupported ter:AudioNotSupported	NVT 不支持音频。

11.7.3 获取兼容音频源配置集

该操作请求获取，与特定媒体配置文件相兼容的设备的所有音频源配置。每个返回的配置应是一个对媒体文件的 AddAudioSourceConfiguration 命令有效的输入参数。结果因设备的功能，配置和设置而不同。支持音频流从 NVT 到客户端的 NVT 支持通过 GetCompatibleAudioSourceConfigurations 命令列出（与特定的配置文件）兼容的音频源配置。

表 156：GetCompatibleAudioSourceConfigurations 命令

GetCompatibleAudioSourceConfigurations	请求-响应
消息名称	描述
GetCompatibleAudioSource-ConfigurationsRequest	包含现有媒体文件的令牌。 tt:ReferenceToken ProfileToken [1][1]
GetCompatibleAudioSource-ConfigurationsResponse	包含于媒体文件兼容的音频源配置清单。 tt:AudioSourceConfiguration Configurations [0][unbounded]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	表明文件令牌 ProfileToken 不存在。
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	NVT 不支持音频。

11.7.4 获取音频源配置选项集

当音频源参数被重新配置是，该操作返回有效的配置选项，如果使用了特定的音频源配置，选项应该关注这些特定的配置。指定的选项应与该媒体文件兼容。支持音频流从 NVT 到客户端的 NVT 支持通过 GetAudioSourceConfigurationOptions 命令列出有效的音频参数选项（对于一个给定的文件和配置）。

表 157：GetAudioSourceConfigurationOptions 命令

GetAudioSourceConfigurationOptions	请求-响应
消息名称	描述
GetAudioSourceConfiguration-OptionsRequest	此消息包含一个音频源可选的令牌配置和媒体文件。 ConfigurationToken 指定一个现有的配置目标选项。 ProfileToken 指定一个选项必须兼容的现有的媒体文件。 tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]
GetAudioSourceConfiguration-OptionsResponse	此消息包含音频的配置选项。指定音频源配置选项应关联特定的配置。指定的选项应与该媒体文件兼容。如果没有指定标记，

	选项应被视为通用设备。 tt:AudioSourceConfigurationOptions Options [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	表明文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	请求的配置不存在。
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	NVT 不支持音频。

11.7.5 修改音频源配置

此操作修改音频源的配置。ForcePersistence 标志表示是否在重新启动 NVT 后保留更改。使用此配置的运行流可立即根据新的设置更新，但所做的更改不会生效，除非客户端请求一个新的 URI 流并重新启动任何受影响的流。如果新的设置无法顺利使用 RTSP，例如通过改变编解码器类型的任何参数，NVT 不使用这些设置给现有的流，相反它必须继续使用旧的设置运行流或对受影响的流停止发送数据。

用 NVC 改变运行流的方法超出本规范的范围。支持从 NVT 到客户端传送音频流的 NVT 支持通过 SetAudioSourceConfiguration 命令配置音频源的参数。

表 158：SetAudioSourceConfiguration 命令

SetAudioSourceConfiguration	请求-响应
消息名称	描述
SetAudioSourceConfiguration-Request	配置元素包含要修改的音频源配置。配置应存在于 NVT。ForcePersistence 元素决定是否在重启后存储和保持配置更改。如果为真，改变是永久的。如果为假，可能会在重新启动后恢复到改变以前的值。 tt:AudioSourceConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]
SetAudioSourceConfiguration-Response	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	配置不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	配置参数无法设置。

env:Receiver ter:Action ter:ConfigurationConflict	新设置与其他使用的设置冲突。
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	NVT 不支持音频。

11.8 音频编码器配置

音频编码器配置包含以下对音频数据进行编码的参数：

- Encoder –用于音频数据的编码。
- Bitrate –输出比特率[kbps]
- SampleRate –输出采样率[kHz]

音频编码器配置结构还包含组播参数和会话超时，用于定义音频流的行为。
如果在配置文件里面使用了一个音频编码配置，该配置的 UseCount 加 1，用于表示改变该配置可能影响其他用户。

11.8.1 获取多个音频编码器配置

此操作列出设备所有现有的音频编码器配置。客户端不需要知道任何有关音频编码器配置就可以使用该命令。支持从 NVT 到客户端传送音频流的 NVT 支持通过 GetAudioEncoderConfigurations 命令列出有效的音频编码器配置。

表 159：GetAudioEncoderConfigurations 命令

GetAudioEncoderConfigurations	请求-响应
消息名称	描述
GetAudioEncoderConfigurations-Request	空消息。
GetAudioEncoderConfigurations-Response	此消息包含 NVT 所有现有的音频编码器配置清单。 tt:AudioEncoderConfiguration Configurations [0][unbounded]
错误码	描述
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	NVT 不支持音频。

11.8.2 获取音频源编码器配置

GetAudioEncoderConfiguration 命令获取音频编码器配置令牌已知的音频编码器配置。支持音频流从 NVT 到客户端的 NVT 支持通过 GetAudioEncoderConfiguration 命令列出一个特定的音频编码器配置。

表 160：GetAudioEncoderConfiguration 命令

GetAudioEncoderConfiguration	请求-响应
消息名称	描述
GetAudioEncoderConfiguration-Request	此消息包含请求的音频编码器配置令牌。 tt:ReferenceToken ConfigurationToken [1][1]
GetAudioEncoderConfiguration-Response	此消息包含请求的音频编码配置的匹配令牌。 tt:AudioEncoderConfiguration Configurations [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	配置不存在。
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	NVT 不支持音频。

11.8.3 获取多个兼容音频编码器配置

此操作要求 NVT 所有的音频编码器配置与特定的媒体文件兼容。每个返回的配置应是媒体文件 AddAudioEncoderConfiguration 命令的有效输入参数。结果因设备的功能，配置和设置而异。支持音频流从 NVT 到客户端的 NVT 支持通过 GetCompatibleAudioEncoderConfigurations 命令列出（与特定的配置文件）兼容的音频编码器配置。

表 161：GetCompatibleAudioEncoderConfigurations 命令

GetCompatibleAudioEncoderConfigurations	请求-响应
消息名称	描述
GetCompatibleAudioEncoder-ConfigurationsRequest	包含现有的媒体文件令牌。 tt:ReferenceToken ProfileToken [1][1]
GetCompatibleAudioEncoder-ConfigurationsResponse	包含一个媒体文件兼容的音频编码器配置清单。 tt:AudioEncoderConfiguration Configurations [0][unbounded]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	请求的文件令牌 ProfileToken 不存在。
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	NVT 不支持音频。

11.8.4 获取音频编码器配置选项集

当音频编码器参数的重新配置时，该操作返回有效的选项。支持音频流从 NVT 到客户端的 NVT 支持通过 GetAudioEncoderConfigurationOptions 命令列出有效的音频编码器（对于一个给定的文件和配置）参数选项。

表 162：GetAudioEncoderConfigurationOptions 命令

GetAudioEncoderConfigurationOptions	请求-响应
消息名称	描述
GetAudioEncoderConfiguration-OptionsRequest	此消息包含选择的音频编码器配置令牌和媒体文件。 ConfigurationToken 指定一个现有配置的目标选项。 ProfileToken 指定一个应兼容现有媒体文件的选项。 tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]
GetAudioEncoderConfiguration-OptionsResponse	此消息包含音频的配置选项。指定的音频编码器配置选项应关联特定的配置。指定的选项应与该媒体文件兼容。如果没有指定标记，选项应被视为通用设备。 tt:AudioEncoderConfi gurationOptions Options [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	配置不存在。
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	NVT 不支持音频。

11.8.5 设置音频编码配置

此操作修改音频编码器配置。ForcePersistence 标志表示在重新启动 NVT 后是否保持更改。组播设置的改变是永久的。使用此配置的运行流可根据新的设置立即更新，但所做的更改不会生效，除非客户端请求一个新的 URI 流并重新启动任何受影响的流。用 NVC 改变运行流的方法超出本规范的范围。支持音频流从 NVT 到客户端的 NVT 支持通过 SetAudioEncoderConfiguration 命令配置音频编码器参数。

表 163：SetAudioEncoderConfiguration 命令

SetAudioEncoderConfiguration	请求-响应
消息名称	描述
SetAudioEncoderConfiguration-Request	配置元素包含修改后的音频编码器配置。配置存在 NVT。 ForcePersistence 元素决定是否在重启后存储和保持配置更改。如果为真，改变是永久的。如果为假，可能会在重新启动后恢复到改变以前的值。 tt:AudioEncoderCo nfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]
SetAudioEncoderConfiguration-Response	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	配置不存在。

env:Sender ter:InvalidArgVal ter:ConfigModify	请求的配置不能设置。
env:Receiver ter:Action ter:ConfigurationConflict	新设置与其他使用的设置冲突。
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	NVT 不支持音频。

11.9 视频分析配置

视频分析配置包含分析引擎和规则引擎的参数（见 4.12 节）。因此，分析引擎由分析服务管理的多个分析模块组成。同样，规则引擎由分析服务管理的多个规则组成。随后命令自动完成视频分析配置。举例来说，ModifyVideoAnalyticsConfiguration 命令自动操作改变分析和规则引擎配置。当视频分析配置是在文件中，元数据配置可以在 RTP 流内激活场景描述流（见第 11.10 节）。

设备可能不允许从多个不同的视频源配置的媒体文件引用非常类似的视频分析配置。如果该设备允许的话，它应为每个配置文件生成单独的场景描述，因为一个场景描述的坐标系统，涉及到一个特定的视频源配置。屏蔽和几何规则也涉及到视频源配置的坐标系统。即使是指同一视频源也可能需要单独为每个视频源配置整个视频分析处理。

由于视频分析配置的选项是动态的且往往由供应商指定的，他们只能通过视频分析服务检索。

11.9.1 获取多个视频分析配置

此操作列出了设备所有的视频分析配置。此命令将列出设备所有配置的视频分析。客户端不需要知道任何关于视频分析就能使用命令。支持视频分析的设备支持通过 GetVideoAnalyticsConfigurations 命令列出有效的视频分析配置。

表 164：GetVideoAnalyticsConfigurations 命令

GetVideoAnalyticsConfigurations	请求-响应
消息名称	描述
GetVideoAnalyticsConfigurations-Request	空消息。
GetVideoAnalyticsConfigurations-Response	此消息包含设备所有现有视频分析的配置清单。 tt:VideoAnalyticsConfiguration Configurations[0][unbounded]
错误码	描述
env:Sender ter:ActionNotSupported ter:VideoAnalyticsNot-Supported	配置不支持视频分析。

11.9.2 获取视频分析配置

GetVideoAnalyticsConfiguration 命令获取视频分析令牌已知的视频分析配置。支持视频分析的设备支持通过 GetVideoAnalyticsConfiguration 命令列出特定的视频分析配置。

表 165：GetVideoAnalyticsConfiguration 命令

GetVideoAnalyticsConfiguration	请求-响应
消息名称	描述
GetVideoAnalyticsConfiguration-Request	此消息包含现有的视频分析配置的令牌。 tt:ReferenceToken ConfigurationToken [1][1]
GetVideoAnalyticsConfiguration-Response	此消息包含请求的视频分析配置。 tt:VideoAnalyticsConfiguration Configurations [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	表示请求配置的配置令牌不存在。
env:Sender ter:ActionNotSupported ter:VideoAnalyticsNot-Supported	设备不支持视频分析。

11.9.3 获取多个兼容视频分析配置

此操作要求设备所有的视频分析配置都和相关的媒体文件兼容。每个返回的配置应是媒体文件 AddVideoAnalyticsConfiguration 命令的有效输入参数。结果因设备的功能，配置和设置而异。支持视频分析的设备支持通过 GetCompatibleVideoAnalyticsConfigurations 命令列出（与特定的配置文件）兼容的视频分析的配置。

表 166：GetCompatibleVideoAnalyticsConfigurations 命令

GetCompatibleVideoAnalyticsConfigurations	请求-响应
消息名称	描述
GetCompatibleVideoAnalytics-ConfigurationsRequest	此消息包含现有的媒体文件令牌。 tt:ReferenceToken ProfileToken [1][1]
GetCompatibleVideoAnalytics-ConfigurationsResponse	包含一个与给定的媒体文件兼容的视频分析配置清单 tt:VideoAnalyticsConfiguration Configurations[0][unbounded]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	请求文件令牌 ProfileToken 不存在。
env:Sender ter:ActionNotSupported ter:VideoAnalyticsNotSupported	设备不支持视频分析。

11.9.4 修改视频分析配置

使用此命令修改视频分析配置。ForcePersistence 标志表示设备在重启后是否保存更改。使用此配置的运行流可根据新的设置立即更新。否则可能由于有非常相似的视频分析配置令牌参考，而在由规则引擎处理的场景描述与分析引擎和规则引擎产生的通知之间出现不一致。支持视频分析的设备支持通过 SetVideoAnalyticsConfiguration 命令配置视频分析参数。

表 167：SetVideoAnalyticsConfiguration 命令

SetVideoAnalyticsConfiguration	请求-响应
消息名称	描述
SetVideoAnalyticsConfiguration-Request	配置元素包含修改后的视频分析配置。配置应存在设备。ForcePersistence 元素决定是否在重启后存储和保持配置更改。如果为真，改变是永久的。如果为假，可能会在重新启动后恢复到改变以前的值。 tt:VideoAnalyticsConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]
SetVideoAnalyticsConfiguration-Response	空消息。
错误码	描述
env:Sender ter:InvalidArgs ter:NoConfig	配置不存在。
env:Sender ter:InvalidArgVal ter:ConfigModify	配置参数无法设置。
env:Receiver ter:Action ter:ConfigurationConflict	新设置与其他使用的设置冲突。
env:Sender ter:ActionNotSupported ter:VideoAnalyticsNot-Supported	配置不支持视频分析。

11.10 元数据配置

元数据配置包含的参数用于选择数据放到在元数据流中的。包括 PTZ 状态、PTZ 位置以及订阅和分析数据定义的事件。事件订阅数据是在第 15.5 节。由分析参数定定义包含来自那一部分文件的分析引擎数据，请参阅 11.9 节。

结构还包含用于配置和控制元数据流组播的组播参数。会话超时参数定义会话超时（见第 12.2.1.1.1 ）

如果配置文件里面使用了一个元数据配置，该配置的 UseCount 加 1，用于表示改变该配置可能影响其他用户。

11.10.1 获取多个元数据配置

此操作列出了所有现有的元数据配置。客户端不需要知道任何元数据的情况就能使用命令。支持元数据流的 NVT 或其他设备应支持通过 GetMetadataConfigurations 命令列出现有的元数据配置。

表 168：GetMetadataConfigurations 命令

GetMetadataConfigurations	请求-响应
消息名称	描述
GetMetadataConfigurations-Request	空消息。
GetMetadataConfigurations-Response	此消息包含设备所有现有的元数据配置清单。 tt:MetadataConfiguration Configurations [0][unbounded]
错误码	描述
	没有具体的错误命令。

11.10.2 获取元数据配置

GetMetadataConfiguration 命令获取元数据令牌已知的元数据配置。支持元数据流的 NVT 或其他设备支持通过 GetMetadataConfiguration 命令列出特定的元数据配置。

表 169：GetMetadataConfiguration 命令

GetMetadataConfiguration	请求-响应
消息名称	描述
GetMetadataConfiguration-Request	此消息包含现有的元数据配置令牌。 tt:ReferenceToken ConfigurationToken [1][1]
GetMetadataConfiguration-Response	此消息包含请求的元数据配置。 tt:MetadataConfiguration Configurations [0][unbounded]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	表明请求的配置令牌 ConfigurationToken 不存在。

11.10.3 获取多个兼容元数据配置

此操作请求设备所有的与相关的媒体文件兼容元数据配置。每个返回的配置应是媒体文件 AddMetadataConfiguration 命令的有效输入参数。结果因设备的功能，配置和设置而异。支持元数据流的 NVT 或其他设备支持通过 GetCompatibleMetadataConfigurations 命令列出（与特定的文件）兼容的元数据配置。

表 170：GetCompatibleMetadataConfigurations 命令

GetCompatibleMetadataConfigurations	请求-响应
消息名称	描述

GetCompatibleMetadata-ConfigurationsRequest	此消息包含现有的媒体文件令牌。 tt:ReferenceToken ProfileToken [1][1]
GetCompatibleMetadata-ConfigurationsResponse	此消息包含与给定媒体文件兼容的元数据配置清单。 tt:MetadataConfiguration Configurations [0][unbounded]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	表明请求文件令牌 ProfileToken 不存在。

11.10.4 获取元数据配置选项集

此操作返回更改元数据配置的有效的选项。支持元数据流的 NVT 或其他设备支持通过 GetMetadataConfigurationOptions 命令列出有效的元数据参数选项（对于一个给定的文件和配置）。

表 171：GetMetadataConfigurationOptions 命令

GetMetadataConfigurationOptions	请求-响应
消息名称	描述
GetMetadataConfiguration-OptionsRequest	此消息包含一个元数据配置可选的令牌和媒体文件。 ConfigurationToken 指定一个现有的配置的目标选项。 ProfileToken 指定一个应兼容现有的媒体文件的选项。 tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]
GetMetadataConfiguration-OptionsResponse	此消息包含元数据的配置选项。指定的元数据配置选项应关联特定的配置。指定的选项应与该媒体文件兼容。如果没有指定标记，选项应被视为通用设备。 tt:MetadataConfigurationOptions Options [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	表明请求文件令牌 ProfileToken 不存在的。
env:Sender ter:InvalidArgVal ter:NoConfig	请求配置不存在。

11.10.5 修改元数据配置

此操作修改元数据配置。ForcePersistence 标志表示设备重启后是否应保持改变。组播设置的改变是永久的。使用此配置的运行流可根据新的设置立即更新。但所做的更改不会生效，除非客户端请求一个新的 URI 流并重新启动任何受影响的流。用 RVC 改变运行流的方法超出本规范的范围。支持元数据流的 NVT 或其他设备支持通过 SetMetadataConfiguration 命令配置元数据参数。

表 172：SetMetadataConfiguration 命令

SetMetadataConfiguration	请求-响应
消息名称	描述
SetMetadataConfiguration-Request	配置元素包含组播设置以及一套在元数据流中确定列入什么样的数据的过滤器。 ForcePersistence 元素决定是否在重启后存储和保持配置更改。如果为真，改变是永久的。如果为假，可能会在重新启动后恢复到改变以前的值。 tt:MetadataConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]
SetMetadataConfiguration-Response	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	请求配置不存在。
env:Sender ter:InvalidArgVal ter:ConfigModify	请求配置不能设置。
env:Receiver ter:Action ter:ConfigurationConflict	新设置与其他使用的设置冲突。

11.11 音频输出

音频输出代表可以连接到扬声器的音频输出端口。

11.11.1 获取音频输出集

此命令列出所有有效的音频输出设备。有一个或多个音频输出端口的 NVT 支持通过 GetAudioOutputs 命令列出有效的音频输出。

表 173：GetAudioOutputs 命令

GetAudioOutputs	请求-响应
消息名称	描述
GetAudioOutputsRequest	空消息。
GetAudioOutputsResponse	包含设备描述的所有有效的音频输出的结构清单。如果一个设备没有音频输出则返回一个空清单。 tt:AudioOutput AudioOutputs [0][unbounded]
错误码	描述
env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	NVT 不支持音频或音频输出。

11.12 音频输出配置

音频输出配置包含以下参数：

- SourceToken: 现有的音频输出参考。
- OutputLevel: 输出音量配置参数。
- SendPrimacy: 一个 NVTs 可以使用的，有半双工音频输入/输出来配置活动的传输方向参数（见第 11.14 节）。

如果配置文件里面使用了一个音频输出配置，该配置的 UseCount 加 1，用于表示改变该配置可能影响其他用户。

11.12.1 获取多个音频输出配置

此命令将列出所有现有设备的音频输出配置。NVC 不需要知道任何有关音频配置就可以使用此命令。在能够输出音频的 NVT 上支持通过这个命令列出音频输出配置。

表 174：GetAudioOutputConfigurations 命令

GetAudioOutputConfigurations	请求-响应
消息名称	描述
GetAudioOutputConfiguration-Request	空消息。
GetAudioOutputConfigurations-Response	包含设备有效的音频输出配置清单。 tt:AudioOutputConfiguration Configurations [0][unbounded]
错误码	描述
env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	设备不支持音频或音频输出。

11.12.2 获取音频输出配置

如果音频输出配置令牌已知，输出配置可以通过 GetAudioOutputConfiguration 命令获取。有一个或多个音频输出的 NVT 支持通过 GetAudioOutputConfiguration 命令检索一个特定的音频输出配置。

表 175：GetAudioOutputConfiguration 命令

GetAudioOutputConfiguration	请求-响应
消息名称	描述
GetAudioOutputConfigurationRequest	此消息包含请求的音频输出配置令牌。 tt:ReferenceToken ConfigurationToken [1][1]
GetAudioOutputConfigurationResponse	此消息包含请求的音频输出配置的匹配令牌。 tt:AudioOutputConfiguration Configurations [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	表示请求配置的 ConfigurationToken 不存在。

env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	设备不支持音频或音频输出。

11.12.3 获取多个兼容音频输出配置

此命令将列出设备所有的与特定媒体文件兼容的音频输出配置。每个返回的配置应是 AddAudioOutputConfiguration 命令有效的输入参数。有一个或多个音频输出的 NVT 支持通过 GetCompatibleAudioOutputConfigurations 命令列出（与特定的文件）兼容的音频输出配置。

表 176：GetCompatibleAudioOutputConfigurations 命令

GetCompatibleAudioOutputConfigurations	请求-响应
消息名称	描述
GetCompatibleAudioOutput-ConfigurationsRequest	包含请求的媒体文件令牌。 tt:ReferenceToken ProfileToken [1][1]
GetCompatibleAudioOutput-ConfigurationsResponse	包含与给定媒体文件兼容的音频输出配置清单 tt:AudioOutputConfiguration Configurations [0][unbounded]
错误码	描述
env:Sender ter:InvalidArgVal ter:No Profile	请求的文件令牌 ProfileToken 不存在。
env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	设备不支持音频或音频输出。

11.12.4 获取音频输出配置选项集

此操作返回配置音频输出有效的选项。有一个或多个音频输出的 NVT 支持通过 GetAudioOutputConfigurationOptions 命令列出有效的音频输出配置选项（对于一个给定的文件和配置）。

表 177：GetAudioOutputConfigurationOptions 命令

GetAudioOutputConfigurationOptions	请求-响应
消息名称	描述
GetAudioOutputConfiguration-OptionsRequest	此消息包含一个音频输出可选的令牌配置和媒体文件 ConfigurationToken 指定一个现有的配置的目标选项。 ProfileToken 指定一个与现有的媒体文件兼容的选项。 tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]
GetAudioOutputConfiguration-OptionsResponse	此消息包含音频输出配置的选项。指定音频输出配置的选项应关联特定的配置。指定的选项应与该媒体文件兼容。如果没有指定标记，选项应被视为通用设备。 tt:AudioOutputConfigurationOptions Options [1][1]
错误码	描述
env:Sender	请求的文件令牌 ProfileToken 不存在。

ter:InvalidArgVal ter:No Profile	
env:Sender ter:InvalidArgVal ter:NoConfig	请求配置不存在。
env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	设备不支持音频或音频输出。

11.12.5 设置音频输出配置

此操作修改音频输出配置。ForcePersistence 标志表示设备重启后是否保持改变。有一个或多个音频输出的 NVT 支持通过 SetAudioOutputConfiguration 命令修改音频输出参数。

表 178：SetAudioOutputConfiguration 命令

SetAudioOutputConfiguration	请求-响应
消息名称	描述
SetAudioOutputConfiguration-Request	设备的配置元素包含修改的音频输出配置。 ForcePersistence 元素决定是否在重启后存储和保持配置更改。如果为真，改变是永久的。如果为假，可能会在重新启动后恢复到改变以前的值。 tt:AudioOutputConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]
SetAudioOutputConfiguration-Response	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	请求配置不存在。
env:Sender ter:InvalidArgVal ter:ConfigModify	配置参数不能设置。
env:Receiver ter:Action ter:ConfigurationConflict	新设置与其他使用的设置冲突。
env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	设备不支持音频或音频输出。

11.13 音频解码器配置

音频解码器配置不包含任何参数配置解码器。解码器对它接收到的每一个数据（根据其能力）进行解码。

如果配置文件里面使用了一个音频解码器配置，该配置的 UseCount 加 1，用于表示改变该配置可能影响其他用户。

11.13.1 获取多个音频解码器配置

此设备列出所有现有的音频解码器配置。NVC 不需要知道任何有关音频解码器配置的情况就可以使用此命令。可以解码音频的 NVT 支持通过此命令列出音频输出配置。

表 179：GetAudioDecoderConfigurations 命令

GetAudioDecoderConfigurations	请求-响应
消息名称	描述
GetAudioDecoderConfigurations-Request	空消息。
GetAudioDecoderConfigurations-Response	包含设备上有用的音频解码器配置清单。 tt:AudioDecoderConfiguration Configurations[0][unbounded]
错误码	描述
env:Receiver ter:ActionNotSupported ter:AudioDecodingNotSupported	设备不支持音频或音频解码。

11.13.2 获取音频解码器配置

如果音频解码器配置令牌是已知的，则可以通过 GetAudioDecoderConfiguration 命令获取解码器配置。一个可以解码音频的 NVT 支持通过 GetAudioDecoderConfiguration 命令检索一个特定的音频解码器配置。

表 180：GetAudioDecoderConfiguration 命令

GetAudioDecoderConfiguration	请求-响应
消息名称	描述
GetAudioDecoderConfigurationRequest	此消息包含请求音频解码器配置的令牌。 tt:ReferenceToken ConfigurationToken [1][1]
GetAudioDecoderConfigurationResponse	包含请求音频解码器配置的匹配令牌。 tt:AudioDecoderConfiguration Configuration [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	请求配置的 ConfigurationToken 不存在。
env:Receiver ter:ActionNotSupported ter:AudioDecodingNotSupported	设备不支持音频或音频解码。

11.13.3 获取兼容音频解码器配置集

此操作列出了设备所有的，与关联媒体文件兼容的音频解码器的配置。每个返回的配置应是媒体文件 AddAudioDecoderConfiguration 命令的有效的输入参数。一个可以解码音频的 NVT 支持通过 GetCompatibleAudioDecoderConfigurations 命令列出（与特定的配置文件）兼容的音频解码器配置。

表 181：GetCompatibleAudioDecoderConfigurations 命令

GetCompatibleAudioDecoderConfigurations	请求-响应
消息名称	描述
GetCompatibleAudioDecoder-ConfigurationsRequest	包含媒体文件令牌。 tt:ReferenceToken ProfileToken [1][1]
GetCompatibleAudioDecoder-ConfigurationsResponse	包含与给定媒体文件兼容的音频解码器配置清单。 tt:AudioDecoderConfiguration Configurations [0][unbounded]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	请求的文件令牌 ProfileToken 不存在。
env:Receiver ter:ActionNotSupported ter:AudioDecodingNotSupported	设备不支持音频或音频解码。

11.13.4 获取音频解码器配置选项集

该命令列出设备的一个给定的文件和配置的音频解码能力。一个可以解码音频的设备支持通过这个命令对音频解码器配置选项的检索。

表 182：GetAudioDecoderConfigurationOptions 命令

GetAudioDecoderConfigurationOptions	请求-响应
消息名称	描述
GetAudioDecoderConfiguration-Options-Request	此消息包含一个音频解码器配置可选的令牌和媒体文件。 ConfigurationToken 指定一个现有的配置的目标选项。 ProfileToken 指定一个与现有的媒体文件兼容的选项。 tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]
GetAudioDecoderConfiguration-Options-Response	此消息包含音频解码器的配置选项。指定的音频解码器配置选项关联特定的配置。指定的选项应与该媒体文件兼容。如果没有指定标记，选项应被视为通用设备。 tt:AudioDecoderConfi gurationOptions Options [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	请求的文件令牌 ProfileToken 不存在。
env:Sender ter:InvalidArgVal ter:NoConfig	请求的配置不存在。
env:Receiver ter:ActionNotSupported ter:AudioDecodingNotSupported	设备不支持音频或音频解码。

11.13.5 设置音频解码器配置

此操作修改音频解码器的配置。ForcePersistence 标志表示设备是否在重启后保存更改。可以解码音频的设备支持通过 SetAudioDecoderConfiguration 命令修改音频解码器参数。

表 183：SetAudioDecoderConfiguration 命令

SetAudioDecoderConfiguration	请求-响应
消息名称	描述
SetAudioDecoderConfiguration-Request	配置元素包含修改音频解码器配置。配置必须存在设备上。ForcePersistence 元素决定是否在重启后存储和保持配置更改。如果为真，改变是永久的。如果为假，可能会在重新启动后恢复到改变以前的值。 tt:AudioDecoderConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]
SetAudioDecoderConfiguration-Response	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	请求的配置不存在。
env:Sender ter:InvalidArgVal ter:ConfigModify	配置参数不能设置。
env:Receiver ter:Action ter:ConfigurationConflict	新设置与使用的配置冲突。
env:Receiver ter:ActionNotSupported ter:AudioDecodingNotSupported	设备不支持音频或音频解码。

11.14 音频通道模式

一个音频通道可支持不同类型的音频传输。全双工操作不需要任何特殊处理，而半双工操作需要切换传输方向。

音频输出配置里面的首选发送参数指示的是目前活动的方向。NVC 可以设置音频输出配置，实现在不同的模式之间切换。

定义首选发送有下列模式：

- www.onvif.org/ver20/HalfDuplex/Server

服务器允许发送音频数据到客户端。客户端不得在这种模式下通过反向通道向 NVT 发送音频数据。

- www.onvif.org/ver20/HalfDuplex/Client

允许客户端通过服务器的反向通道发送音频数据。NVT 不得在这种模式下向客户端发送音频数据。

- www.onvif.org/ver20/HalfDuplex/Auto

它是由设备决定如何处理发送和接收音频数据。

取消声学回声超出 ONVIF 的范围。

11.15 URI 流

11.15.1 获取Uri流

该操作请求一个 URI，该 URI 可以被用来启用一个使用 RTSP 作为控制协议的实时媒体流。返回 URI 应无限期地保持有效，即使文件被改变。ValidUntilConnect、ValidUntilReboot 和 Timeout 参数应当相应地设置（ValidUntilConnect = false，ValidUntilR EBOOT = false，Timeout = PT0S）。NVT 应该支持通过 GetStreamUri 命令，对一个具体的媒体配置文件，检索一个媒体流 URL。

与其他 ONVIF 的服务完全兼容的设备不应该产生超过 128 个字节的 URI。

表 184：GetStreamUri 命令

GetStreamUri	请求-响应
消息名称	描述
GetStreamUriRequest	StreamSetup 元素包含两个部分。流类型规定一个单播或组播的流媒体是否请求。运输指定一个链传输协议确定在不同的网络协议的流媒体的通道。 ProfileToken 元素表示使用媒体文件去定义流的内容的配置。 tt:StreamSetup StreamSetup [1][1] tt:ReferenceToken ProfileToken [1][1]
GetStreamUriResponse	包含被请求的流媒体使用的稳定的 URI 以及参数定义的完整的 URI。ValidUntilConnect 和 ValidUntilReboot 参数设置为假，Timeout 参数应设置为 PT0S，此 URI 流是无限期有效，即使文件改变。 xs:anyURI Uri [1][1] xs:boolean InvalidAfterConnect [1][1] xs:boolean InvalidAfterReboot [1][1] xs:duration Timeout [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	媒体文件不存在。
env:Sender ter:InvalidArgVal ter:InvalidStreamSetup	不支持 StreamSetup 的流类型或传输部分的规范。

env:Sender ter:OperationProhibited ter:StreamConflict	StreamSetup 的流类型或传输部分的规范与其他流冲突。
env:Receiver ter:Action ter:IncompleteConfiguration	指定媒体文件不包含任何未使用的来源或编码器配置。

11.16 快照

11.16.1 获取Uri快照

网络客户端使用 GetSnapshotUri 命令从 NVT 获得 JPEG 快照 。返回的 URI 应无限期地保持有效，即使文件改变。ValidUntilConnect、ValidUntilReboot 和 Timeout 参数应相应的设置（ValidUntilConnect = FALSE，ValidUntilReboot = FALSE，Timeout = PT0S）。URI 可通过一个 HTTP GET 操作获取 JPEG 图像。图像编码总是 JPEG 的编码设置，忽略在媒体配置文件中的编码设置。NVT 支持此命令。

表 185：GetSnapshotUri 命令

GetSnapshotUri	请求-响应
消息名称	描述
GetSnapshotUriRequest	ProfileToken 元素表示使用媒体文件并将定义源和快照的尺寸。 tt:ReferenceToken ProfileToken [1][1]
GetSnapshotUriResponse	包含被请求的流媒体使用的稳定的 URI 以及参数定义的完整的 URI。ValidUntilConnect 和 ValidUntilReboot 参数设置为假，Timeout 参数应设置为 PT0S，此 URI 流是无限期有效，即使文件改变。 xs:anyURI Uri [1][1] xs:boolean InvalidAfterConnect [1][1] xs:boolean InvalidAfterReboot [1][1] xs:duration Timeout [1][1]
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	媒体文件不存在。
env:Receiver ter:Action ter:IncompleteConfiguration	指定媒体文件不包含任何视频编码器配置或视频源配置。

11.17 组播

NVT 和客户端组播流的详细讨论见 12.1 节。

11.17.1 开始组播流

此命令使用 NVT 的指定媒体文件开始组播流。流持续直至相同的文件呼叫停止组播流。

NVT 重启后流将持续直到收到一个停止组播流请求。组播地址，端口和 TTL 分别在视频编码配置、音频编码器配置和元数据配置里分别设置。支持视频、音频或元数据组播流的 NVT 支持通过 StartMulticastStreaming 命令启动一个组播流。

表 186：StartMulticastStreaming 命令

StartMulticastStreaming	请求-响应
消息名称	描述
StartMulticastStreamingRequest	包含用来定义组播流的文件令牌。 tt:ReferenceToken　ProfileToken [1][1]
StartMulticastStreamingResponse	空消息。
错误码	描述
env:Sender 　ter:InvalidArgVal 　　ter:NoProfile	媒体文件不存在。
env:Receiver 　ter:Action ter:IncompleteConfiguration	指定的媒体文件不包含一个引用视频编码视频源的配置到音频源或音频编码配置或元数据配置的参考。

11.17.2　停止组播流

此命令停止组播流使用 NVT 上的媒体文件。支持视频，音频或元数据的 NVT 支持通过 StopMulticastStreaming 命令停止指定的组播流。

表 187：StopMulticastStreaming 命令

StopMulticastStreaming	请求-响应
消息名称	描述
StopMulticastStreamingRequest	包含用来定义组播流的文件令牌。 tt:ReferenceToken　ProfileToken [1][1]
StopMulticastStreamingResponse	空消息。
错误码	描述
env:Sender 　ter:InvalidArgVal 　　ter:NoProfile	媒体文件不存在。
env:Receiver 　ter:Action ter:IncompleteConfiguration	指定的媒体文件不包含一个引用视频编码或视频源的配置到音频源或音频编码配置或元数据配置的参考。

11.18 同步点

11.18.1　设置同步点

同步点允许客户端进行准确解码同步点后的所有数据。

例如，如果一个视频流被配置一个大的 I-帧距离且客户端丢失一个单包，则客户端不显示视频，直到下一个 I-帧传输。在这种情况下，客户可以强制要求 NVT 尽快添加一个 I-帧的

同步点。客户可以要求文件的同步点。NVT 应为文件添加所有相关联的数据流的同步点。

同样，通过元数据流，同步点用于更新完整的 PTZ 或事件的状态。

一个 I-帧应加入到与文件相关的视频流。如果一个事件流与文件相关，同步点应请求处理，（如 15.6 节所述）。如果 PTZ 元数据流与文件相关，PTZ 位置应在元数据流内重做。

支持 MPEG -4 或 H.264 的 NVT 支持通过 SetSynchronizationPoint 命令请求 I-帧。

表 188：SetSynchronizationPoint 命令

SetSynchronizationPoint	请求-响应
消息名称	描述
SetSynchronizationPointRequest	包含请求的同步点的文件参考。 tt:ReferenceToken ProfileToken [1][1]
SetSynchronizationPointResponse	空消息。
错误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	文件不存在。

11.19服务具体的错误码

下表列出了媒体服务的具体错误代码。此外，每个命令还可以生成一个通用的错误，见表 6。

具体错误是作为一个通用的错误子码被定义的，请参阅第 5.11.2.1 。母公司通用的子码是在下面每一行的顶部的子码，具体的错误子码是在表格的底部。

表 189：具体的媒体服务错误码

错误码	母码 子码	错误原因	描述
env:Receiver	ter:ActionNotSupported ter:AudioNotSupported	没有音频功能	NVT 不支持音频。
env:Receiver	ter:Action ter:MaxNVTProfiles	已达最大数	已达到支持文件的最大数。
env:Receiver	ter:ActionNotSupported ter:AudioOutputNotSupported	没有音频输出能力	NVT 不支持音频或音频输出。
env:Receiver	ter:ActionNotSupported ter:AudioDecodingNotSupported	没有音频解码能力	NVT 不支持音频或音频解码。
env:Receiver	ter:Action ter:IncompleteConfiguration	设置不完整	指定文件缺失实体要求。
env:Receiver	ter:Action ter:ConfigurationConflict	使用新设置时冲突	新设置与其他使用的设置冲突
env:Sender	ter:InvalidArgVal ter:ProfileExists	文件令牌已存在	有 ProfileToken 文件的已存在。

env:Sender	ter:InvalidArgVal ter:NoConfig	配置令牌不存在	请求的配置令牌 ConfigurationToken 不存在。
env:Sender	ter:InvalidArgVal ter:NoProfile	文件令牌不存在	请求的文件令牌 ProfileToken 不存在。
env:Sender	ter:Action ter:DeletionOfFixedProfile	固定的文件不能被删除	固定的文件不能被删除。
env:Sender	ter:InvalidArgVal ter:ConfigModify	不能设置参数	配置参数不能设置。
env:Sender	ter:ActionNotSupported ter:VideoAnalyticsNotSupported	没有视频分析能力	NVT 不支持视频分析。
env:Sender	ter:InvalidArgVal ter:InvalidStreamSetup	无效的流安装	不支持流类型规格或 StreamSetup 的传输部分。
env:Sender	ter:OperationProhibited ter:StreamConflict	流冲突	流类型规范或在 StreamSetup 的传输部分与其他流冲突。

12 实时流

这一章将对实时视频流，实时音频流以及元数据流进行讲解。但是在这一章中不会介绍与实时流相关具体的服务。可以通过在 Media Service 和 ReceiverService 中定义的网页服务命令来进行实时配置。

12.1 流媒体协议

12.1.1传输格式

实时传输协议（RTP）是一个媒体传输协议（参考 12.1.2）。接下来的 4 节将描述 RTP 数据的传输。

12.1.1.1 通过 UDP 的 RTP 数据传输

UDP 能以一种有效的方式来传输实时数据，并且具有最小的开销。设备应该支持 RTP/UDP 协议并且同样的支持 RTP/UDP 组播。

12.1.1.2 通过 TCP 传输 RTP 数据

如果通过 UDP 传输有包丢失，那么协议标准允许通过另外一种媒体传输方式 TCP 来传输 RTP 数据。设备可以基于选择性的支持 RTP/TCP 协议。如果设备支持 RTP/TCP 协议，那么这协议应该符合[RFC 4571] (面向连接的帧实时传输协议，面向连接 RTP 控制协议包传输)。

12.1.1.3 RTP/RTSP/TCP

设备应该支持使用 RTP/RTSP 协议来进行媒体传输；并且 RTP/RTSP 协议在传输过程中，使用 RTSP 隧道来穿越防火墙。这一协议应当符合[RFC 2326 标准]（参见 10.12 节）。

12.1.1.4 RTP/RTSP/HTTP/TCP

为了穿越防火墙，数据流应当通过 HTTP 发送。设备应当支持通过 RTP/RTSP/HTTP/TCP 协议来传输媒体。如果设备支持 TLS1.0，那么为了穿越防火墙，数据流应当通过 HTTPS 来进行发送以及接收，以及一个设备应当支持通过 RTP/RTSP/HTTP/TCP 协议来进行媒体传输。协议应当符合[RFC 2326](第 12.2.1.1 节 RTSP：嵌入（交错）2 进制数据).。

(穿越防火墙)隧道的方法也应该符合 QuickTime，这 QuickTime 可以从苹果公司获取。以下文件命令的部分应该被 NVT 提供。

http://developer.apple.com/quicktime/icefloe/dispatch028.html

12.1.2 媒体传输

12.1.2.1 RTP

这 RTP 协议是提供对两个终端之间流媒体进行实时传输的协议。RTP 协议支持排序（序列号），去抖动（时间戳）和媒体同步（同步源表示符）。

所有通过 RTP 协议传输的媒体流应当符合[RFC 3550], [RFC 3551],[RFC 3984], [RFC 3016]以及通过 RTP 传输的 JPEG 相关规定（参见 12.1.3）。

<table>
<tr><td colspan="7">0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</td></tr>
<tr><td>V</td><td>P</td><td>X</td><td>CC</td><td>M</td><td>PT</td><td>序列号</td></tr>
<tr><td colspan="7">时 间 戳</td></tr>
</table>

同步源标识符

图 14：RTP 头

一个 RTP 头可以用以下值进行填充。

表 190：RTP 头值

头字段	值	描述
版本（V）：2bits	2	
填充（P）：1bit	0/1	如果有效载荷包含了填充的八位，那么这应该设置为 1
扩展（X）：1bit	0/1	取决于对 RTP 头的扩展使用；使用 RTP 头扩展来传递附加信息可以分为两种情况来：1.通过 RTP 传输 JPEG（参见 12.1.3 节） 2.重播 如果使用了扩展，那么扩展位应该被设置
CSRC 计数（CC）:4bit	0	
标志（M）：1bit	0/1	这种方法应当符合所有 RFC 相关的规定（(e.g. [RFC3984] for H.264 Video）或者达到这些标准（eg：“JPEG over RTP”见 12.1.3）或者 RTP 元数据流的(参见 12.1.2.1.1)
负载类型（PT）：7bits	参见第六章的[RFC 3551]	
序列号：16bits		这“sequence number”初始值应当是随机以及不可预知的；这样能够让已知明文攻击更加的困难 每发送一个 RTP 数据包时这值就增 1
时间戳: 32 bits		时间戳的初始值应当是随机以及不可预测的，这样能够让已知明文的攻击更加困难 更详细的规则请查看 12.1.2.2.1 节媒体同步 对时间戳的使用还取决于编码器

同步源 32 bits		对于数据流的同步时钟源，协议并没有对之做出相关的限制

12.1.2.1.1 RTP 元数据流

元数据流也能够通过 RTP 协议进行传输。对于元数据传输的 RTP 协议头中的负载类型，标识以及时间戳，可以通过以下方式来对它们进行定义：

在一个 RTSP 会话建立的过程中，应该对负载类型分配一个动态的负载类型值（96-127）。

当 XML 文件关闭时，这 RTP 标记位应该置 1；

推荐使用一个时钟频率为 90000HZ 的时间戳来表示 RTP 包的产生时间。在元数据流中只有 UTC 时间戳会被用到。视频与音频数据流同步是通过 RTCP 协议来控制。

元数据负载是一个拥有根节点的 XML 文件 tt:MetaDataStream。对于 XML 文件，这里没有对其大小限制。当对于流有一个同步点的请求时，这先前的 XML 文档应该关闭，然后开始一个新的文件 XML 文件。推荐在一秒后开始新的 XML 文档（这是最长的时间）。元数据流的 RTP 时间戳没有具体的含义；元数据流复用来自不同来源的元数据。本协议分别对视频分析的场景描述、 云台云台控制器的状态和事件配置的通知都定义了占位符。设备能够在媒体的配置时选择里面的哪些部分应该被复用进元数据（参见 11.10 节）。在文件中，每一个部分都可以以任意顺序在文件中出现多次。一个元数据通过反向通道机制进行双向连接（参见 12.3 节）。

元数据流包含以下元素：

1.视频分析流

2.云台流

3.事件流

对于不同来源元数据的占位符都是以以下 XML 结构：

```
<xs:complexType name="VideoAnalyticsStream">
  <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:element name="Frame" type="tt:Frame"/>
    ...
  </xs:choice>
</xs:complexType>

<xs:complexType name="PTZStream">
  <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:element name="PTZStatus"/>
    ...
  </xs:choice>
</xs:complexType>

<xs:complexType name="EventStream">
  <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:element ref="wsnt:NotificationMessage"/>
    ...
  </xs:choice>
</xs:complexType>
```

以下是一个元数据 XML 文档的例子

```
<?xml version="1.0" encoding="UTF-8"?>
<tt:MetaDataStream xmlns:tt="http://www.onvif.org/ver10/schema">
  <tt:VideoAnalytics>
    <tt:Frame UtcTime="2008-10-10T12:24:57.321">
      ...
    </tt:Frame>
    <tt:Frame UtcTime="2008-10-10T12:24:57.621">
      ...
    </tt:Frame>
  </tt:VideoAnalytics>
</tt:MetaDataStream>

<?xml version="1.0" encoding="UTF-8"?>
<tt:MetaDataStream xmlns:tt="http://www.onvif.org/ver10/schema">
 <tt:Event>
 <wsnt:NotficationMessage>
 <wsnt:Message>
 <tt:Message UtcTime= "2008-10-10T12:24:57.628">
 ...
 </tt:Message>
 </wsnt:Message>
 </wsnt:NotficationMessage>
 </tt:Event>
 </tt:MetaDataStream>
```

12.1.2.2 RTCP

这 RTP 控制协议提供对 RTP 提供服务的质量反馈以及同步不同媒体流。这 RTCP 协议应该符合[RFC 3550].

对于一个反馈的请求，设备应该支持[RFC 4585] 和 [RFC 5104]

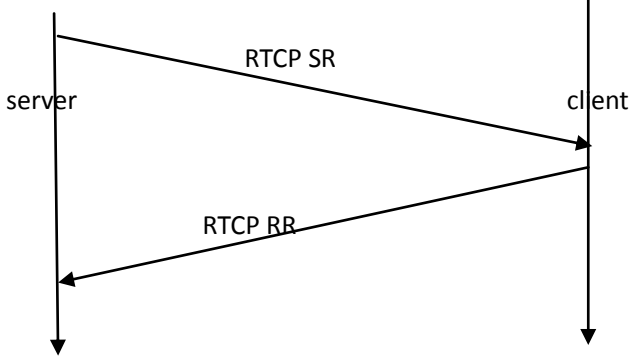


图 15：RTCP 序列

12.1.2.2.1 媒体同步

客户可以从超过一个以上设备同时接受音频和视频流。在这种情况下，每一种流使用不同的时钟（从数据采集到包接收）。RTCP 发送报告（SR）被用来同步不同的媒体流。RTCP SR 应该符合[RFC3550]

这 RTCP 发送报告（SR）包包含了 RTP 时间戳以及一个挂钟时间戳（绝对日期和时间，64 位网络时间协议）。参考 16 章

一个设备应该支持 RTCP 发送报告来同步媒体。客户端也应该使用 RTCP 来同步媒体流。

0	1	2	3
0 1 2 3 4 5 6 7	8 9 0 1 2 3 4 5	6 7 8 9 0 1 2 3 4 5 6 7 8 9	0 1

V	P	RC	PT=SR=200	LENGTH
SSRC of sender				
NTP timestamp, most significant word				
NTP timestamp, least significant word				
RTP timestamp				
sender's packet count				

图 16：RTCP 发送报告

挂钟应该在设备中是通用的，以及每一个时间戳的值应该确定是正确的值。客户端应该用基于 RTP 时钟和挂钟时间戳的合适时间来同步不同媒体（见图 17）。

在有多种设备的情况下，这 NTP 时间戳应该对所有设备是通用的，以及能够在系统中请求 NTP 服务。

图 17：媒体同步

12.1.3 同步点

同步点允许客户在同步点之后编码以及更正使用的数据。在出现错误编码(或者包丢失)情况时，客户可以申请一个同步点来促使设备尽可能快的增加一帧；以及完成对当前云台和事件状态的请求，客户都可以申请一个同步点来解决。

另外，设备应该支持在基于 11.18.1 节和 15.6 节所讲的基本网络服务方法，这个标准推荐使用在[RFC4585]中的 PL1 信息来获取一个同步点

12.1.4 通过RTP传输JPEG

12.1.4.1 所有包的结构

根据【RFC 2435】的规则来传输 JPEG 流。像下面指定的那样：通过选择 RTP 头的扩展，可以在一些 RTP 数据包中嵌入【RFC2435】限制的数据。然而，这种选择改变了包含这样数据包的帧的确切语义。

图 18 显示了 JPEG RTP 包的所有格式。

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1		
根据 RFC 3500 的标准 RTP 头		
0xFFD8 / 0xFFFF	扩展长度	

扩展有效负载： 附加 JPEG 标记段的序列 用 0XFF 填充到总的延伸长度
根据 RFC2435 的 RTP/JPEG 头
熵编码的扫描数据段

图 18：RTP/JPEG 包结构

为了能够从其他可能的头扩展中区分任意一个 RTP 头扩展；在一帧的初始包中，这最先的 16bits((四字节扩展头的前两个字节))应该是 0XFFD8 而一帧的其它 RTP 包的值是 0XFFFF 。

正如[RFC 3550]要求那样，任意存在的头扩展应该通过 RTP 头 X 位进行标记；在扩展头范围内扩展长度，这扩展长度用扩展负载后的 32 位的进行表示。例如， 32 位的扩展头后是一个 0 长度区域，那么这就表示是一个空的头扩展 。

熵编码的扫描数据段不是在所有的 RTP 包中存在。然而，一个完整的 RTP/ JPEG 头应在每帧的初始数据包和所有包含熵编码的扫描数据段的包中存在，否则可能会丢失。

根据[RFC 2435]，在 RTP/JPEG 头中，如果不存在头的扩展，那么碎片的偏移区域将被使用。 另外，对于一个已经传输的的帧中，如果所有的包都包含一个熵编码扫描数据段，或者没有一个包包含一个熵编码扫瞄数据段但是这些包包含头扩展，那么碎片偏移应该不为 0。根据这个协议，如果帧的初始化包中没有头扩展，那么它的碎片偏移区域不应该为 0，否则应该设为 0。根据 3.1.8 节的【RFC2435】规定，凡是包括 0 偏移的的碎片 RTP/JPEG 头和一个在 128-255 之间的 Q 值的包都应该包含量化表头，而其它的包不应该包含这个报头。

12.1.4.2 逻辑解码规范

对于解码规范，它是假设在 RTP 流原始数据包的顺序已根据 RTP 序列编号恢复。

如果帧的初始数据包不包含上述规定 RTP 头扩展，解码器应产生完整的扫描头和按照[RFC 2435] 进行解码。扫描数据段和任何遵照这个协议的头扩展应以在流中发生的顺序（忽略碎片偏移值）进行连接，直到下一个有位设置的 RTX 包到来时结束。

否则（至少有一个像上面所讲的空的头扩展存在在这初始化的包中），下列规则适用于每一个这样的帧：

1.如果帧的初始数据包不包含一个熵编码的扫描数据段，但包含一个像上面讲的那样的头扩展，那么解码器对遵从这个子序列包协议的头扩展负载进行连接，直到包含有 RTP 标记位的或者包含熵编码的第一个包出现。

2.对有 JPEG SOI 标记的 RTP 头扩展负载连接时，有这些标记应该在理论上优先出现。

3.在一个帧的初始化包中，如果这 RTP/JPEG 扫描头的 Q 值不为 0，那么量化表应该根据[RFC 2435]进行预初始化。如果 Q 值等于 0，那么子序列帧的量化表都应该从先前帧进行拷贝，在初始化头扩展负载时允许帧忽略 DQT 标记。

4.如果初始化 RTP 头扩展负载支持没有 DRI 标记，但是这帧的初始化包中的 RTP/JPEG 头包含了一个 RTP/JPEG 重启标记，那么一个符合【RFC2435】规则的 DRI 标记就会被添加进初始化头扩展负载中。另外，如果这初始化 RTP 头扩展支持 DRI 标记，那么应该根据【RFC2435】规则在相同帧中优先重启。然而，如果重启间隔被使用，为了对只遵守【RFC2435】规则的编码器兼容，编码通常应该在 RTP/JPEG 重启标记赋合适的值。

DRI 的标记位不应该从先前帧获取。
如果初始化的 RTP 头扩展的负载（序列）支持没有 SOF 标记，否则，那样会优先执行，一个 SOF 标记应该用以下值进行添加：

1.如果 RTP/JPEG 的宽度和高度区域为 0，应使用先前帧的 SOF 标记位。
2.否则这标记应该根据【RFC2435】规则来分配。

然而，只要图像的大小在【RFC2435】指定的变化范围，编码器应该在 RTP/JPEG 头中指定与 SOF 值相符合图形的大小。
如果这初始化的头扩展负载支持没有 SOS 标记，应该根据【RFC2435】规则来得到一个相应 SOS 标记以及在信息中附加上去，相反的这扩展的 SOS 标记会优先执行。

一个 SOS 标记不应从先前的帧获得。
如果在扩展区域 SOS 标志是存在并且不是在熵编码扫描数据之后，标志将在一帧初化扩展负载(序列)内的最后的标记。 用 0xFF 进行的必要填充不会跟随这个标志，但是可以优先于它。

保存的熵编码扫瞄数据和头扩展负载应该按在 RTP 流中发生的顺序进行逻辑添加，直到遇到 RTP 标记位时结束。如果在帧的逻辑序列中，一个 EOI 标记也应该在最后添加。

对于每帧，在生成序列在遇到第一个（可能被添加）EOI 标记时才是有效（可能被缩写）的 JPEG 流，从对帧解码程序中得出一个的完整图像。这就意味着在每一帧第一个 EOI 标记以后的数据是协议规范之外的事情。

12.1.4.3 支持的彩色空间和采样因素

发射机应该采用只灰度等级和 YCbCr 彩色空间。 客户将支持灰度等级和 YcbCr。
对于 YCbCr 的采样因素应该符合 [RFC 2435]支持的值。 例如：4:2：0 (首选)或 4:2 :2.

12.1.4.4 像素长宽比处理

JPEG 文件像素长宽比可以通过 JFIF 标志指定。 如果像素长宽比与根据[RFC 2435]标准 1:1 和 1:2 比不相同，那么这个标记应该在每帧的初始头扩展负载中(序列)指定像素点长宽比。

12.1.4.5 隔行扫描处理

隔行视频扫描是在两个独立场进行编码以及在 RTP/JPEG 中按【RFC2435】指定进行标记。
两个场将使用同样的彩色空间，采样因素和像素长宽比
如果帧是逐行扫描的，那么就不应该使用隔行编码。。

12.2 媒体控制协议

12.2.1流控制

使用 URI 中定义的协议来控制媒体流。对定义在第 11.15.1 节的 GetStreamUri 命令进行响应并返回一个 URL

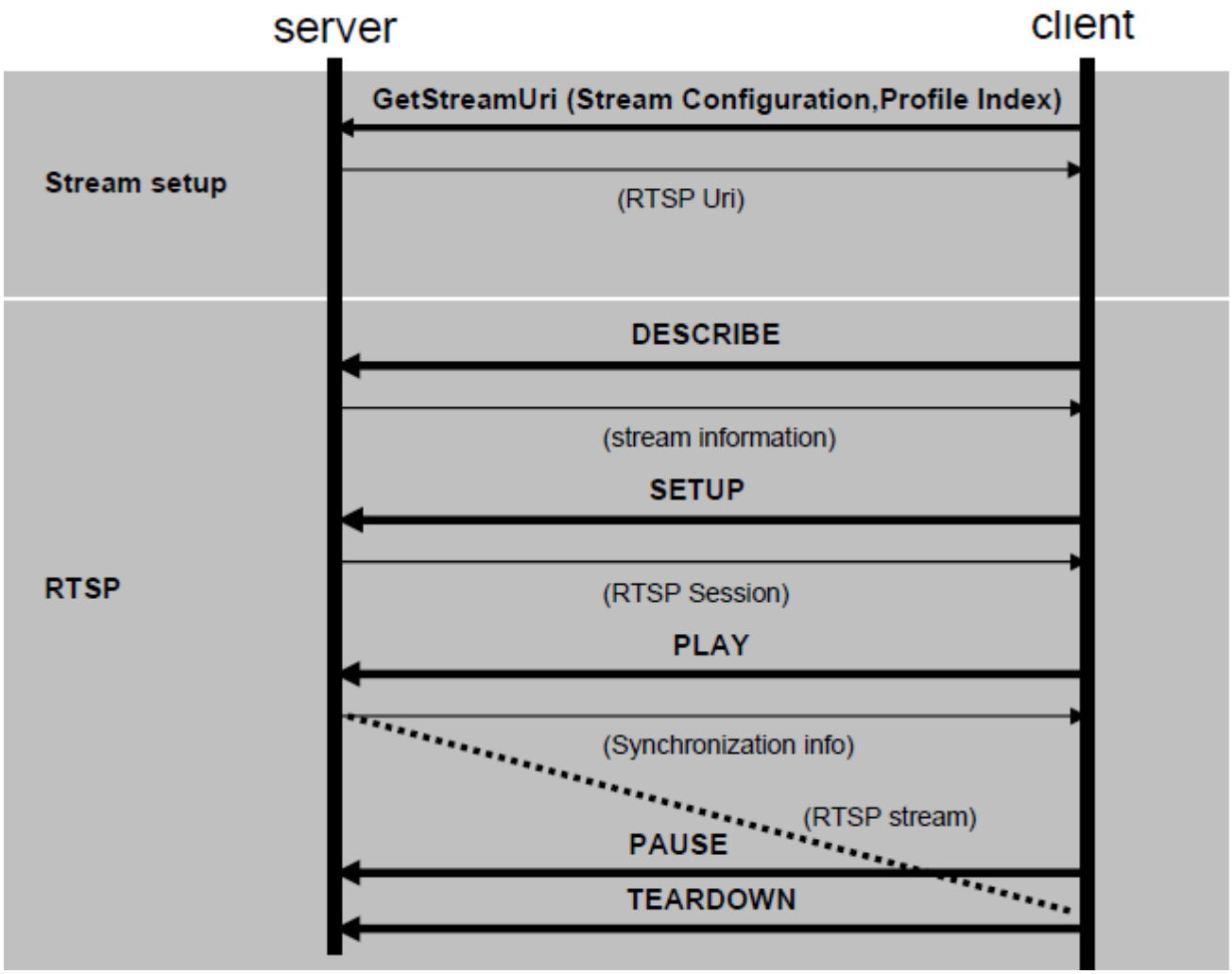


图 19：流控制

12.2.1.1 RTSP

所有设备以及客户端都应当支持 RTSP 协议([RFC 2326])来进行初始化会话以及回放的控制。 RTSP 应使用 TCP 协议作为其传输协议，RTSP 传输默认的 TCP 端口 554。会话描述协议（SDP）应该被用来提供流媒体信息以及 SDP 协议应该符合【RFC4566】标准 。

表 191：RTSP 方法

方法	方向	SPEC	描述
OPTIONS	R->T T->R	M X	可在任意时刻发出 OPTIONS 请求，如用户打算尝试非标准请求，并不影响服务器状态
DESCRIBE	R->T	M	请求在设计文件中检索媒体参数
ANNOUNCE	R->T T->R	X	
SETUP	R->T	M	请求设置媒体会话参数
PLAY	R->T	M	请求开始媒体流
PAUSE	R->T	O	请求暂停媒体流： 在一个狭窄的带宽中处理多个流媒体，通过挂载 RTP 流，通过减少多余数据能

			够控制网络拥塞
TEARDOWN	R->T	M	请求释放媒体会话
GET_PARAMETER	R->T T->R	O	
SET_PARAMETER	R->T T->R	O O	用任意一个可选的方法来让一个 RTSP 会话保持活跃
REDIRECT	T->R	X	
RECORD	R->T	X	

12.2.1.1.1 保持 RTSP 会话的方法

一个 RTSP 客户端保持 RTSP 会话，并防止它从会话超时结束（见 RFC2326）第 12.37 ）。本规范推荐以下的方法来保持的 RTSP 单播和组播的流：

1.客户可以选择使用 Set<configurationEntity>EncoderConfiguration 命令来设置超时参数；命令定义在 11.2 节，否则超时值会默认设置为 60

2.在所有的 RTSP SETUP 响应，发射机应包括根据[RFC 2326]第 12.37 定义超时值和发射机应该使用超时值的来保持会话

3.为了保持 RTSP 会话，客户端应通过任何 RTSP 方法或者发或发送 RTCP 接收器报告。推荐使用 SET_PARAMETER 命令。

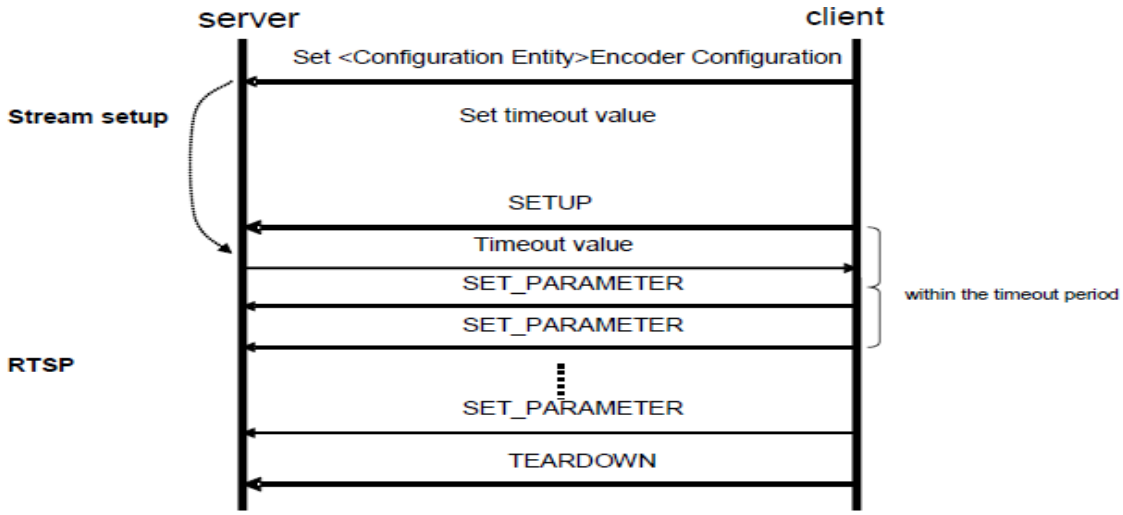


图 20：保持会话

12.2.1.1.2 RTSP 音频和视频同步

为了使客户端可以立即开始同步视频和音频流，对传进来进行录制的包计算出的绝对 UTC 时间戳；这些传输进来的包是用来存储的。一个发射机应在对 RTSP PLAY 应答信息包含以下头区域 RTSP 播放响应头字段：

1. Range ([RFC 2326] section 12.29) 。时钟单元应该包含一个起始时间（[RFC 2326]第 3.7 节），not SMPTE or NPT units.。

2. RTP-Info ([RFC 2326] section 12.33)。这应该包含一个与 Range 相关的 RTPTIME 值。

```
Example:
client->server:    PLAY rtsp://example.com/onvif_camera/video RTSP/1.0
                   CSeq: 4
                   Range: npt=now-
                   Session: 12345678

server->client:          RTSP/1.0 200 OK
                   CSeq: 4
                   Session: 12345678
                   Range: 20100217T143720.257Z-
                   RTP-Info: url=rtsp://example.com/onvif_camera/video;
seq=1234;rtptime=3450012
```

12.2.1.1.3 元数据流的 RTSP 会话

在元数据流的情况下，这 SDP 协议描述的“application”应该被用来描述对媒体类型的应答上以及“vnd.onvif.metadata”应该用来对名字编码。

例如 RTSP 协议描述一个 RTSP 服务器（Server）和一个 RTSP 客户端（客户端）进行信息的交换 ：

```
      client->server:  DESCRIBE rtsp://example.com/onvif_camera RTSP/1.0
                       CSeq: 1

  server->client:      RTSP/1.0 200 OK
                       CSeq: 1
                       Content-Type: application/sdp
                       Content-Length: XXX
                       v=0
                       o=- 2890844256 2890842807 IN IP4 172.16.2.93
                       s=RTSP Session
                       m=audio 0 RTP/AVP 0
                       a=control:rtsp://example.com/onvif_camera /audio
                       m=video 0 RTP/AVP 26
                       a=control:rtsp://example.com/onvif_camera /video
                       m=application 0 RTP/AVP 107
                       a=control:rtsp://example.com/onvif_camera/metadata
                       a=recvonly
                       a=rtpmap
                       a=rtpmap:107 vnd.onvif.metadata/90000
```

12.2.1.1.4 RTSP 消息的例子

这个例子显示了信息在 RTSP 客户端和一个 RTSP 服务器进行传递。客户端从设备请求一个音频和视频流。使用 GetStreamUri 命令能够检索一个 URL “ rtsp :/ / example.com / onvif_camera”。参阅第 11.15.1

```
client->server:    DESCRIBE rtsp://example.com/onvif_camera RTSP/1.0
                   CSeq: 1

server->client:    RTSP/1.0 200 OK
                   CSeq: 1
                   Content-Type: application/sdp
                   Content-Length: XXX
                   v=0
                   o=- 2890844256 2890842807 IN IP4 172.16.2.93
                   s=RTSP Session
                   m=audio 0 RTP/AVP 0
                   a=control:rtsp://example.com/onvif_camera/audio
                   m=video 0 RTP/AVP 26
                   a=control:rtsp://example.com/onvif_camera/video

client->server:    SETUP rtsp://example.com/onvif_camera/audio RTSP/1.0
                   CSeq: 2
                   Transport: RTP/AVP;unicast;client_port=8002-8003

server->client:    RTSP/1.0 200 OK
                   CSeq: 2
                   Transport: RTP/AVP;unicast;client_port=8002-8003;
                              server_port=9004-9005
                   Session: 12345678; timeout=60

client->server:    SETUP rtsp://example.com/onvif_camera/video RTSP/1.0
                   CSeq: 3
                   Transport: RTP/AVP;unicast;client_port=8004-8005
                   Session: 12345678

server->client:    RTSP/1.0 200 OK
                   CSeq: 3
                   Transport: RTP/AVP;unicast;client_port=8004-8005;
                   server_port=9006-9007
                   Session: 12345678; timeout=60

client->server:    PLAY rtsp://example.com/onvif_camera RTSP/1.0
```

```
                    CSeq: 4
                    Range: npt=now-
                    Session: 12345678

server->client:     RTSP/1.0 200 OK
                    CSeq: 4
                    Session: 12345678
                    RTP-Info: url=rtsp://example.com/onvif_camera/video;
                    seq=1234;rtptime=3450012,
                    url=rtsp://example.com/onvif_camera/audio;
                    seq=22434;rtptime=1234566

client->server:     TEARDOWN rtsp://example.com/onvif_camera RTSP/1.0
                    CSeq: 5
                    Session: 12345678

server->client:     RTSP/1.0 200 OK
                    CSeq: 5
                    Session: 12345678
```

12.2.1.2 通过 HTTP 的 RSTP

为了穿越防火墙，基于 HTTP / HTTPS 的 RTSP 协议应该被设备支持。见第 12.1.1.4 的 RTP / RTSP / HTTP / TCP 协议。

12.3 往回通道连接

本节介绍如何在客户端和服务器之间建立一个双向的连接。使用 RTSP 协议[RFC 2326]对反向通道连接进行处理。这里将存在一种机制用于表明客户端想建立一个往回通道连接。RTSP 提供了功能标签来处理这样的附加的功能。

支持双向连接的设备（如音频或元数据连接）应该支持已经介绍 RTSP 扩展。

12.3.1 RTSP协议请求的标签

[RFC 2326]的 RTSP 标准，可以用附加对象头进行扩展。为了这个目的，将引进一个请求的标签来处理专门的附加功能（见 RFC2326] ， 1.5 扩展 RTSP 和 12.32 需求）
请求标签是用来判断是否支持此功能。这头应包含任何请求并且要求服务器理解这些特性并能够正确对这些请求进行应答。

支持反向通道的一种装置，应当理解的反向通道的标签：

www.onvif.org/ver20/backchannel

希望建立具有数据反向通道的 RTSP 连接的客户应该在请求信息中包括请求的 Require 头

12.3.2双向连接的连接设置

客户端应当在它的描述请求中包括特性标签，用以表明应该建立双向数据连接。
理解这一要求标签的服务器应在 SDP 协议中包含一个正如在媒体类别的配置一样的额外流媒体。

根据的 RTSP 标准，一个不理解的反向通道的特性标签或不支持双向数据连接 RTSP 服务器会响应错误代码 551 Option not supported。然后，客户端可以尝试建立一个没有反向通道 RTSP 连接。

一个 SDP 的文件被用来描述会话。服务器应在每一个媒体选项中包括一个 a= sendonly 或 b= recvonly 属性来表明将要发送数据的方向。

服务器应该列举出所有支持的解码规则作为自己媒介选项以及客户应该选择哪一种。

12.3.2.1 例一：没有往回支持的服务

```
Client - Server:          DESCRIBE rtsp://192.168.0.1 RTSP/1.0
                          CSeq: 1
                          User-Agent: ONVIF Rtsp client
                          Accept: application/sdp
                          Require: www.onvif.org/ver20/backchannel

Server - Client:          RTSP/1.0 551 Option not supported
                          CSeq: 1
                          Unsupported: www.onvif.org/ver20/backchannel
```

12.3.2.2 例二：使用 ONVIF 往回通道支持的服务

```
Client - Server:          DESCRIBE rtsp://192.168.0.1 RTSP/1.0
                          CSeq: 1
                          User-Agent: ONVIF Rtsp client
                          Accept: application/sdp
                          Require: www.onvif.org/ver20/backchannel

Server - Client:          RTSP/1.0 200 OK
                          CSeq: 1
                          Content-Type: application/sdp
                          Content-Length: xxx

                          v=0
                          o= 2890842807 IN IP4 192.168.0.1
                          s=RTSP Session with audiobackchannel
                          m=video 0 RTP/AVP 26
                          a=control:rtsp://192.168.0.1/video
                           a=recvonly
                           m=audio 0 RTP/AVP 0
                          a=control:rtsp://192.168.0.1/audio
                          a=recvonly
                          m=audio 0 RTP/AVP 0
                          a=control:rtsp://192.168.0.1/audioback
                          a=rtpmap:0 PCMU/8000
```

这 SDP 文件完全描述 RTSP 会话。服务器给客户端控制 URL 来设置流。
在接下来的几步客户能够设置会话：

```
Client - Server:          SETUP rtsp://192.168.0.1/video RTSP/1.0
                          CSeq: 2
                          Transport: RTP/AVP;unicast;client_port=4588-4589

Server - Client:    RTSP/1.0 200 OK
                          CSeq: 2
                          Session: 123124;timeout=60
                          Transport:RTP/AVP;unicast;client_port=4588-4589;
                          server_port=6256-6257

Client - Server:          SETUP rtsp://192.168.0.1/audio RTSP/1.0
                          CSeq: 3
                          Session: 123124
                          Transport: RTP/AVP;unicast;client_port=4578-4579

Server - Client:          RTSP/1.0 200 OK
                          CSeq: 3
                          Session: 123124;timeout=60
                          Transport:RTP/AVP;unicast;client_port=4578-4579;
                          server_port=6276-6277

Client - Server:          SETUP rtsp://192.168.0.1/audioback RTSP/1.0
                          CSeq: 4
                          Session: 123124
                          Transport: RTP/AVP;unicast;client_port=6296-6297
                          Require: www.onvif.org/ver20/backchannel

Server - Client:          RTSP/1.0 200 OK
                          CSeq: 4
                          Session: 123124;timeout=60
                          Transport:RTP/AVP;unicast;client_port=6296-6297;
                          server_port=2346-2347
```

第三步请求建立音频反向通道连接。
在接下来的几步中客户端通过 PLAY 请求开始会话.

```
Client - Server:          PLAY rtsp://192.168.0.1 RTSP/1.0
                          CSeq: 5
                          Session: 123124
                          Require: www.onvif.org/ver20/backchannel

Server - Client:          RTSP/1.0 200 OK
                          CSeq: 5
                          Session: 123124;timeout=60
```

在接收到对 PLAY 请求进行 OK 应答后，客户端可以开始发送音频数据到服务器。收到应答之前，它不会发送数据给服务器在服务器。

请求头表明一个与 PLAY 命令相关的解释是必要的。命令涵盖既从 NVT 到客户端的视频和音频流的起点以及开始从客户机到服务器的音频连接。
为了终止会话，客户端发送一个 TEARDOWN 请求。

```
     Client - NVT:        TEARDOWN rtsp://192.168.0.1 RTSP/1.0
                          CSeq: 6
                          Session: 123124
                          Require: www.onvif.org/ver20/backchannel

     NVT - Client:        RTSP/1.0 200 OK
                          CSeq: 6
                          Session: 123124
```

12.3.3组播流

如果客户准备在组播发送的数据，在 SETUP 请求的参数告诉服务器的多播地址和端口。

12.3.3.1 例：多播设置

```
Client - Server:          SETUP rtsp://192.168.0.1/audioback RTSP/1.0
                          CSeq: 4
                          Session: 123124
                          Transport:RTP/AVP;multicast;destination=224.2.1.1;port=60
                          000-60001;ttl=128
                          Require: www.onvif.org/ver20/backchannel

Server - Client:          RTSP/1.0 200 OK
                          CSeq: 4
                          Session: 123124;timeout=60
                          Transport:RTP/AVP;multicast;destination=224.2.1.1;port=60
                          000-60001;ttl=128;mode="PLAY"
```

12.4 错误处理

RTSP 和 HTTP 协议错误被分为不同的类别（例如，状态代码 1XX ， 2XX ， 3XX ， 4xx 和 5xx ） 。设备和客户端应当支持处理这些状态代码。在 RTSP 与[RFC 2326]相关的状态代码定义是第 11.0。对于 HTTP 状态代码的定义是指]第 10.0 节 HTTP/1.1 的[2616。

13 接收端配置

这项服务提供了一些管理接收对象的命令（用于从其他设备接收媒体流的命令）。接收器包含如何建立媒体流，接收器的工作模式，以及媒体流 URI（通用资源标志符）这些信息。一个设备应支持至少 128 字节长度的媒体资源标识符。一个接收设备最大的 RTSP 资源标识符的长度的能力显示了设备支持的最大长度。接收服务应该可以由接收媒体流的设备来完成。

在发送设备资源标识符中，赋予这个接收设备的 IP 和 DNS 地址是托管接收服务设备用于访问服务发送设备的地址。举个例子，如果客户端不得不通过一个路由器与发射设备和接收设备来交流，给接收设备的发射设备地址（这种情况是本地网络地址）可能与客户端访问发射设备的地址不同（这种情况是外部网络地址）。

通过实时 RTP 协议（看 12.1.1.1 节），一个设备应该可以支持 RTP 传输，并且通过 RTSP/HTTP/TCP 传输协议，也可以实现 RTP 传输。一个设备可以支持其他的 RTP 传输协议，同时它还应该用适当的能力表明支持什么。（看在表 11 中接收器的种类）

13.1 持久性

所有创建在接收机服务之内的对象都应该持久，它们要再断电重启后保存不变，同样的，所有在对象中的配置数据也应该持久。

13.2 接收端模式

一个接收器可以用三种不同的模式操作：

一直连接。接收器试图保持持续与配置端点的连接。

不连接。接收器不尝试连接

自动连接，接收器根据用户所需进行连接。

13.3 接收命令

本节的描述命令是为接收服务提供的。

13.3.1 获得多个接收器

此操作列表列出了目前存在的设备中所有接收器，接收服务应该支持这个命令。

表 192: 获得接收器的命令

获得接收器	
消息名字	说明
获得设备请求	消息空
获得设备的应答	包含一个列表的接收机
误码	说明
没有具体的故障码	

13.3.2 获得单个接收器

此操作用来搜索具体的接收器（令牌将被客户端识别的接收器）
接收器服务支持这个命令

表 193：getreceiver 命令

GetReceiver	
信息名字	描述
GetReceiver 请求	包含令牌请求的接收者。receivertoken
GetReceiver 应答	包含详细的要求接收机
误码	详细
env:Sender ter:InvalidArgVal ter:UnknownToken	接收器靠 receivertoken 得显示不存在

13.3.3 创建接收器

这个操作创建了一个新的接收器，接收器服务应该支持这个命令。

表 194 ： 创建接收器命令

创建接收器	
信息名字	描述
创建接收器的请求	包含这个接收器的初始配置 tt:ReceiverConfiguration Configuration [1][1]
创建接收器的应答	包含详细的被创建的接收器 tt:Receiver Receiver [1][1]
误码	描述
env:Sender	指定的配置无效

ter:InvalidArgVal ter: InvalidStreamSetup	
env:Receiver ter:Action ter:MaxReceivers	支持的最大数量的接收机已达到

13.3.4 删除接收器

此操作删除现有接收器，一个接收机不能删除它如果正在使用中，本接收器应该支持这个命令

表 195：删除接收器的命令

删除接收器	
信息名字	描述
删除接收器请求	包含标记的接收器被删除
删除接收器应答	信息空
误码	描述
env:Sender ter:InvalidArgVal ter:UnknownToken	接收机靠 receivertoken 得显示不存在
env: Receiver ter:Action ter:CannotDeleteReceiver	删除指定的接收是不可能的，例如因为它是目前使用的。

13.3.5 配置接收器

这个操作是配置接收机，接收机服务应该支持这个命令

表 196：配置接收机

配置接收机	
信息名字	描述
配置接收机请求	包括令牌的接受接受和新配置 tt:ReceiverToken ReceiverToken [1][1] tt:ReceiverConfiguration Configuration [1][1]
配置接受的应答	消息空
误码	描述
env:Sender ter:InvalidArgVal ter:UnknownToken	接收器靠 receivertoken 得显示不存在
env:Sender ter:InvalidArgVal ter:BadConfiguration	指定的配置无效

13.3.6 设计接收器模式

此操作可能被用来确定模式的接收器独立于其配置，接收器服务支持这个命令

表 197：设置接收机模式命令

设置接收机模式	
信息名字	描述
设置接收机模式请求	包含令牌的请求的接收和新模式 tt:ReceiverToken ReceiverToken [1][1] tt:ReceiverMode ReceiverMode [1][1]
设置接收机应答	消息空
误码	描述
env:Sender ter:InvalidArgVal ter:UnknownToken	接收器通过 receivertoken 的显示不存在

13.3.7 获取接收机状态

这一操作可以决定是否目前的接收机断开，连接　或者正在试图连接。接收机应该支持这一命令。

表 198：获得接收机状态命令

获得接收机状态	
消息名字	描述
获得接收机状态请求	包含令牌请求的接收者 tt:ReceiverToken ReceiverToken [1][1]
获得接收机状态应答	包含当前状态的接收机 tt:ReceiverState State [1][1]
误码	描述
env:Sender ter:InvalidArgVal ter:UnknownToken	接收器通过 receivertoken 的显示不存在

13.4 事件

接收机服务通过事件服务来调度事件。它还应能触发事件（无论什么时候在本章列出的条件都能触发）

13.4.1 改变状态

无论什么时候改变接收机的状态，设备应迅速处理以下事件：

Topic: tns1: Receiver/ChangeState

<tt:MessageDescription IsProperty="false">

<tt:Source>

<tt:SimpleItemDescription Name="ReceiverToken" Type="tt:ReceiverToken"/>
</tt:Source>
<tt:Data>
<tt:SimpleItemDescription Name="NewState" Type="tt:ReceiverState"/>
<tt:SimpleItemDescription Name="MediaUri" Type="tt:MediaUri" minOccurs="0"/>
</tt:Data>
</tt:MessageDescription>

13.4.2 连接失败

如果一个接收机不能建立一个连接，设备应迅速处理以下事件：

Topic: tns1: Receiver/ConnectionFailed
<tt:MessageDescription IsProperty="false">
<tt:Source>
<tt:SimpleItemDescription Name="ReceiverToken" Type="tt:ReceiverToken"/>
</tt:Source>
<tt:Data>
<tt:SimpleItemDescription Name="MediaUri" Type="tt:MediaUri"/>
</tt:Data>
</tt:MessageDescription>

13.5 服务器错误码

表 199 列出了显示服务具体的误码。此外，每个命令都可以生成一个通用的误码，看表 6.

具体的故障被定义为一个通用的故障，看 5.11.2.1 部分。这个双亲通用码在每一行子码的上面，具体的故障子码是在底部的基本单元。

表 199：服务指定的误码

误码	双亲通用码 通用码	产生误码的原因	描述
Env:Sender	ter:InvalidArgVal ter:UnknownToken	接收机不存在	接收机靠 ReceiverToken 得暗示不存在
Env:Sender	ter:Action ter:MaxReceivers	已经达到了接收机的最大数量	能支持接收机的最大数量已经达到了
Env:Sender	ter:InvalidArgVal	不支持流设置	流型号的规范或者接收机配置流设置的传输部

	ter:InvalidStreamSetup		分不支持
Env:Sender	ter:Action	不可能删除这些设备	不可能删掉指定的设备，比如目前正在使用
	ter:CannotDeleteReceiver		

14 显示服务

显示设备有一个固定数量的视频输出，每一个都可以连接到监控器，通过使用设备输入输出服务，客户端可以请求设备的视频输出。每一个这些输出都配置了布局(比如单个视图和多画面).布局定义了一定数量的视频窗格，每个视频窗格都占据着一个物理显示区域。

一个网络的视频显示同时可能有一定数量的音频输入和音频输出，每一个这些输出可能都与一个窗格联系起来的。将一个音频输入或音频输出和窗格地图对应联系起来，可以让来自传输设备的音频和视频流自动的正确的输出。窗格还包含了一个对应接收器的标记(链接已经存储了的，从显示设备到发射机的一些必要的信息)

显示服务提供配置窗格功能而且可以自动描述和改变这个设备的布局。可能要求到一些布局和视频输出的编码能力。

一个显示设备应该支持显示服务就像[DisplayService.wsdl]中定义的那样。

14.1 窗格

一个窗格在监空器里面是一个显示区域，它连接到视频输出。每个窗格都有一个窗格配置来说明究竟是哪个实体连接到了窗格。这个窗格配置包括：

窗格令牌：在显示装置中一个唯一的标示符。

窗格名字：配置名字

音频输出令牌：音频输出的指针与窗格联系起来。客户端可以检索一个设备的现有的音频输出，通过设备输入输出服务获得音频输出命令。

音频来源令牌：音频来源的指针与此窗格联系在一起的。通过反馈机制，显示设备到 NVT 的音频连接已经建立起来。用户可以检索音频源设备的输入输出服务的获得设备音频来源命令。

音频编码器配置：音频编码器配置包括编码器和解码器，比特率和采样率。

接收机令牌：一个接收机的指针有一些用来接收发射机数据的必需的信息。接收机可以被连接起来，并且网络视频显示器在指定输出的地方显示接收来的数据。通过使用接收机服务的获得接收机命令，用户可以检索可用的接收机。用户必须通过设置音频输出和音频源令牌建立连接来配置窗格。如果一个令牌没有被设置，相应的会话将不能建立起来。

改变一个窗格的配置或一个参考接收机的参数不会影响 RTSP 协议的连接。如果一个客户端打算申请一个新的参数，它将会重启 RTSP 的连接。

视频输出窗格的布局决定了窗格是否是可见的，在哪里可见的。如果窗格是可见的，接收机可以只建立一个 RTSP 连接来接收数据。布局的改变对于正在传输的流不起作用。

14.1.1 获得多个窗格配置

对于一个指定的视频输出，这个命令列出了当前一个设备中所有确定的的窗格（无论这个窗格此时是否可见），通过此命令，这个显示设备应该支持检索它自己配置的窗格。

表 200：获得窗格配置

获得窗格配置	请求-应答
信息名字	描述
获得窗格配置请求	该视频输出要素指定了视频输出（对应的窗格配置被要求了）
获得窗口配置应答	包括了一系列的指定的视频输出定义的的窗格，每一个视频输出至少有一个窗格配置， tt:PaneConfiguration PaneConfiguration [1][unbounded]
误码	描述
env:Sender ter:InvalidArgVal ter:NoVideoOutput	被要求的视频输出不存在

14.1.2 获得单个窗格配置

如果窗格令牌已经被这个能用来获得窗格配置的命令识别的话，那么通过此命令，这个设备应该支持检索特定的窗格配置

表 201：获得窗格配置

获得窗格配置	请求-应答
信息名字	描述
获得窗格配置请求	这个信息包含了一个窗格（配置已经被要求了）的令牌也包含了一个视频输出令牌（包含请求窗格的指定视频输出）这个消息还包含请求令牌的窗格 tt:ReferenceToken VideoOutput[1][1] tt:ReferenceToken Pane[1][1]
获得配置窗格应答	包含请求的窗格配置 tt:PaneConfiguration PaneConfiguration [1][1]

误码	描述
env:Sender ter:InvalidArgVal ter:NoPane	请求的窗格不存在
env:Sender ter:InvalidArgVal ter:NoVideoOutput	要求的视频输出不存在

14.1.3 设置多个窗格配置

这个命令可以用一步改变所有存在的音频输出的窗格配置。消息包括了所有的窗格配置（修改或未修改）。显示设备应该支持这个改变窗格配置通过这个命令。

表 202：设置窗格配置

设置窗格配置	请求-应答
设置窗格配置请求	这一消息包含视频输出的令牌和新的窗格配置
设置窗格应答	消息空
误码	描述
env:Sender ter:InvalidArgVal ter:NoPane	要求的窗格不存在
env:Sender ter:InvalidArgVal ter:NoVideoOutput	要求的视频输出不存在
env:Sender ter:InvalidArgVal ter:InvalidConfig	配置不可能设置
env:Sender ter:InvalidArgVal ter:NoVideoOutput	请求的视频输出不存在

14.1.4 设置单个窗格配置

此命令用来改变单个窗格的配置， 显示设备可以支持单个窗格配置的修改，通过此命令。

设置单个窗格配置	请求-应答
信息名字	描述
设置单个窗格配置请求	这个消息包含视频输出和新窗格配置的令牌 tt:ReferenceToken VideoOutput[1][1] tt:PaneConfiguration PaneConfiguration[1][1]

设置单个窗格配置应答	消息空
误码	描述
env:Sender ter:InvalidArgVal ter:NoPane	请求窗格不存在
env:Sender ter:InvalidArgVal ter:NoVideoOutput	请求视频输出不存在
env:Sender ter:InvalidArgVal ter:InvalidConfig	配置不可能被设置

14.1.5 创建窗格配置

这个命令用于创建一个窗格配置。通过此命令，一个支持窗格动态创建的显示设备应该支持动态窗格配置的创建

表 204：创建窗格配置

创建窗格配置	请求-应答
信息名字	描述
创建窗格配置请求	这个消息包含了视频输出的令牌和新的窗格配置 tt:ReferenceToken VideoOutput[1][1] tt:PaneConfiguration PaneConfiguration[1][1]
创建窗格配置请求	消息空
误码	描述
env:Sender ter:InvalidArgVal ter:MaxNumberOfPane	达到了最大的窗格量
env:Sender ter:InvalidArgVal ter:NoVideoOutput	请求的视频输出不存在
env:Sender ter:InvalidArgVal ter:InvalidConfig	这个配置不能设置

14.1.6 删除窗格配置

这个命令用于删除一个窗格配置。通过此命令，一个支持动态窗格删除的显示设备应该支持动态窗格配置的删除。

表 205：删除窗格配置

创建窗格配置	请求-应答
信息名字	描述
删除窗格配置请求	这个消息包含了视频输出的令牌和新的窗格配置 tt:ReferenceToken VideoOutput[1][1] tt:PaneConfiguration PaneConfiguration[1][1]
删除窗格配置请求	消息空
误码	描述
env:Sender ter:InvalidArgVal ter:MaxNumberOfPane	不可能删除这个窗格配置
env:Sender ter:InvalidArgVal ter:NoVideoOutput	请求的视频输出不存在
env:Sender ter:InvalidArgVal ter:InvalidConfig	被请求的窗格配置不存在

14.2 布局

布局为每个窗格配置分配了一点区域来显示，布局的设置直接影响一个具体的视频输出。这个布局包括一系列的窗口配置和与它们相联系的显示区域。如果设备支持重叠的窗格，那么窗格显示在显示器中的顺序就被窗格配置清单的顺序确定下来了。清单中的第一个窗格在前景中显示出来。一个设备要么提供可以一个固定的布局要么它可以自由的配置布局。

14.2.1 获得布局

此命令返回当前视频输出的布局，一个显示设备应支持检索其布局，通过这个命令。

表 206：获得布局

获得布局	要求-应答
信息名字	描述

获得布局要求	包含连接到输出显示的视频输出令牌 tt:ReferenceToken VideoOutput[1][1]
获得布局应答	包含目前视频输出的布局 tt:Layout Layout[1][1]
误码	描述
env:Sender ter:InvalidArgVal ter:NoVideoOutput	被要求的视频输出不存在

14.2.2 设置布局

设置布局操作可以用来改变显示的布局。通过这个命令，显示设备应该支持布局的改变

表 207：设置布局

设置布局	要求-应答
信息名字	描述
设置布局要求	这一消息包含视频输出的令牌和修改过的布局
设置布局应答	消息空
误码	描述
env:Sender ter:InvalidArgVal ter:InvalidLayout	不可能设置布局
env:Sender ter:InvalidArgVal ter:NoVideoOutput	要求的视频输出不存在

14.3 显示选项

设备的显示选项包括支持布局（布局选项），解码，编码能力。获得显示命令返回所有布局，编解码能力。

布局选项

这个布局选项描述了一个设备固定的和预定义的布局。如果这个设备没有提供固定的布局，或者允许自由的设置布局，那么这个要素为空。

编码能力

通过使用合适的算法，网络视频显示可以编码音频和视频流。根据制造商的选择，网络视频显示提供了一些音频和视频的编码器，比特率，分解 。
为了确保不同设备之间的互操作性，本规范给下面的简写的解码器分配了不同的任务：
NVD 应该支持 JPEG QVGA
NVD 将支持 G.711μ Law (如果它还支持视频)
对 NVT ，这些则是相同的强制性的编码器。
一个解码器没有任何参数来配置。一个解码器要解码所有它接收到的内容。万一出现解码错误，解码器应该试着请求一个同步点，并且继续解码。就像 14.4 章节定义那样，这应该是件很正常的事情。编解码能力要素显示了一个设备的解码和编解码能力。

14.3.1 获取显示选项

这个命令列出了视频输出的布局和编解码能力。通过这个命令，一个显示设备应该支持显示选项的检索。

表 208：获得显示选项

获得显示选项	请求-应答
信息名字	描述
获得显示选项请求	有一个指示什么媒体文件要被删除的窗格令牌。
获得显示应答	这个消息包含一个视频输出令牌（这些选项是为之准备的）
误码	描述
env:Sender ter:InvalidArgVal ter:NoVideoOutput	请求的输出令牌不存在

14.4 事件

通过事件服务，显示设备可以调度事件

14.4.1 解码错误事件

当设备收到一个不能解码的比特流的时候，它应该能够产生下列事件：
一个解码器为什么不能解码比特流有多种原因，下面定义的误码可以用来通知客户对于解码错误
1）“不支持的编解码器和不支持的编解码器简介”- 设备不能用来编码比特流，因为这个设备不支持编解码或简介。通过设备的编码能力，客户应该试着重新配置传输机。
2）“数据包错误”— 在比特流中有丢失的或的不期望的数据包。
其他供应商特定的代码也可以

Topic: tns1:VideoDecoder/DecodingError
<tt:MessageDescription IsProperty="false">

<tt:Source>
<tt:SimpleItemDescription Name="VideoOutputToken"
Type="tt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:SimpleItemDescription Name="PaneReference"
Type="tt:ReferenceToken"/>
<tt:SimpleItemDescription Name=”Error” Type=”xs:string”
minOccurs=”0”/>
</tt:Data>
</tt:MessageDescription>

14.5 服务错误码

表 209 列出了显示设备具体的误码。除此之外，每个命令也可以生成一个通用的故障，看表 6.

具体的故障是作为一个通用的故障码被定义的。看 5.11.2.1 章节。双亲子码在下面每一行的顶部，它也是字码，并且具体错误的字码在每个单元的底部

表 209 服务指定的误码

误码	双亲字码 字码	错误原因	描述
env:Sender	ter:InvalidArgVal ter:NoVideoOutput	视频输出不存在	被请求的视频输出不存在
env:Sender	ter.InvalidArgVal ter:FixPane	固定的窗格配置	不能删除这些窗格配置
env:Sender	ter:InvalidArgVal ter:MaximumNumberOfPanes	达到了窗格的最大量	达到了窗格最大量因此不能创建窗格
env:Sender	ter:InvalidArgVal ter:NoPane	窗格配置不存在	被请求的窗格配置不存在
env:Sender	ter:InvalidArgVal ter:InvalidConfig	不能设置窗格配置	被请求的配置不被设备支持

env:Sender	ter:Action ter:InvalidLayout	不能设置布局	被请求的布局不被设备支持

15 事件处理

一个事件是一个行为或事件检测（一个客户可以订阅的设备进行的事件检测）。事件处理是通过事件服务来实现的。一个设备应该提供一个事件服务，就像[ONVIF Event WSDL]里描述的那样。NVT 和 NVC 都应该支持[WS-Addressing]为事件服务。

在这个标准中处理事件是基于[WS-BaseNotification] and [WS-Topics]规范的。本标准要求了基本通知接口实施（implementation），就像在 15.1 中描述的那样，这完全符合[ws-basenotification]的规范。此外，该设备应该实现实时的通知 Pull-Point 接口和通知流接口，就像 15.2 和 15.3 章节定义的那样。

本标准介绍了一个通知消息扩展，从而允许客户端通过事件的性质跟踪对象。属性定义在章节 15.4

事件描述的有效性和在订阅范围之内的过滤在章节 15.5 中讨论。章节 15.6 描述了如何通过客户使用三个通知接口之一请求同步点。章节 15.7 描述了一体化的主题，章节 15.9 描述了处理错误。

最后一部分详细演示了（包括过滤消息和主题设置的通知接口）的实时拉取点通知接口的使用。在相应的[WS-BaseNotification]规范中可以找到一些基本通知接口的例子。

15.1 基本通知接口

章节 15.1.1 简要介绍基本的通知接口的[ws-basenotification]规范。章节 15.1.2 总结了强制性和可选的接口[ws-basenotification]规范

15.1.1 介绍

在通知模式中，以下的逻辑实体参与了，

客户端：实现通知消费者的接口

事件服务：实现通知生产者接口

订阅管理：实现 BaseSubscriptionManager 接口

在一个设备中事件服务和订阅管理都应该被实体化。

典型的实体之间交换消息的顺序显示在图 21. 首先，客户端建立一个到事件服务的连接。客户端可以通过发送一个订阅请求来认证一个通知。如果事件服务接受预订，它代表订阅动态的实例一个订阅管理。事件服务应当返回一个订阅管理者的 WS 终端地址到订阅请求中。为了传输通知来匹配订阅另一个对客户的事件服务开始建立。通过这个建立，事件服务发送一个单向通知消息给客户端的 notificationconsumer（消费者通知）接口。相应的通知可以随时发送服务给客户，当订阅是积极的（active）。

为了控制这个订阅，客户端直接订阅管理者的地址应该由订阅应答返回。在订阅请求中，客户可以指定一个终止时间。该订阅管理者自动注销当终止时间达到时。更新请求可以由客户端发起，以推迟终止时间。客户也可以明确地终止订阅管理者，通过发送退订请求。当成功退订后，订阅管理者不再存在。

事件服务和订阅管理者之间得相互作用不再被[WS-BaseNotification]进一步规定，并且由设备确定。

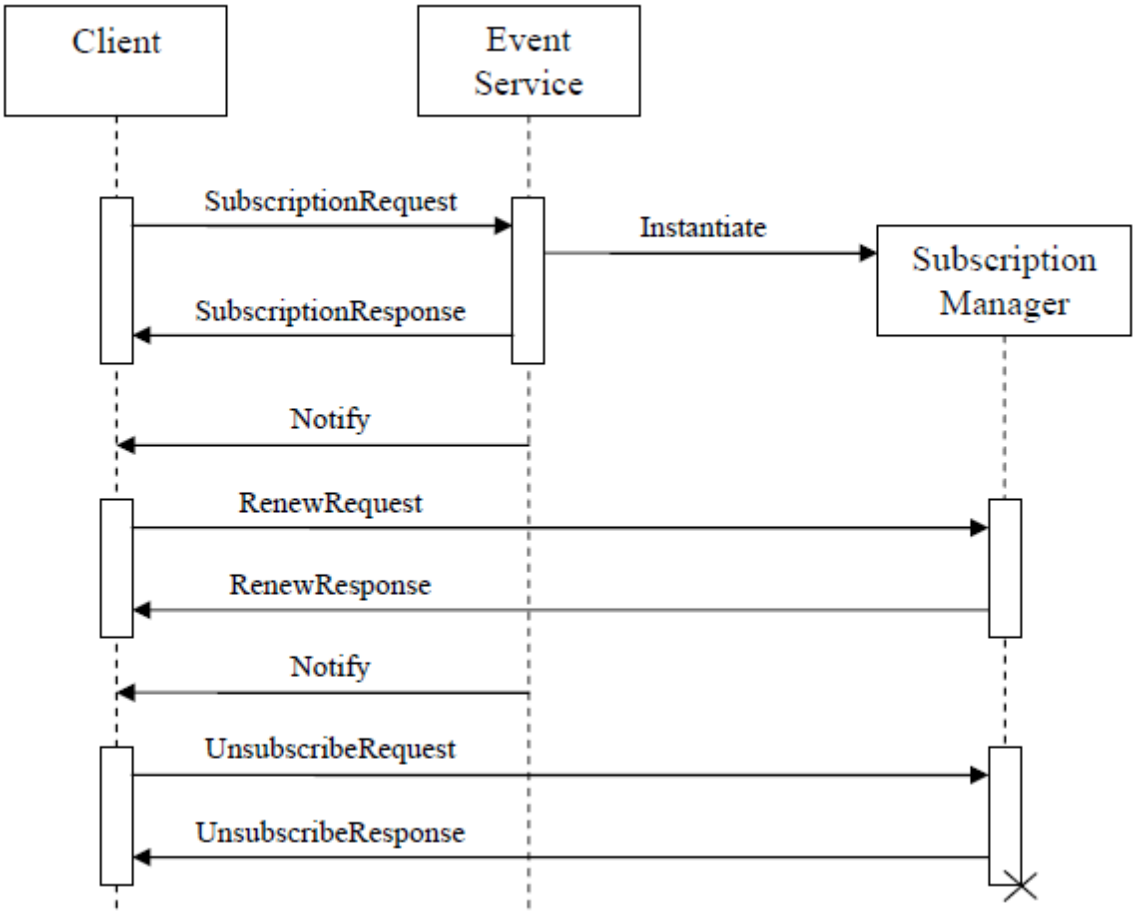


图 21：序列图为基本的通知接口

15.1.2 要求

本节详细说明设备应该提供的 WS 基本通知接口

一个 ONVIF 兼容设备应该支持[WSBaseNotification]通知产生接口。结果，the NotificationProducer Resource Properties 是 OPTIONAL（看 15.5 部分）设备应该至少支持 TopicExpression 和 MessageContent 过滤器，部分 15.5.5 和 15.7.3 中描述的那样。如果设备不接受 InitialTerminationTime 的订阅，在故障码以内，它将提供一个有效的 InitialTerminationTime。在[WS-BaseNotification]规范中，设备应该能够使用 Notify wrapper 给出通知。对设备来说 SubscriptionPolicy wsnt:UseRaw 是一个可选的。在一个通知请求中，虽然[wsbasenotification]具有实时性并且 terminationtime 作为可选的元素，一个 ONVIF 兼容设备也应列出它们。本设备可能用故障码回应任何 GetCurrentMessage 请求，暗示在这个请求中，目前没有消息是可用的。

在设备上一个[WS-BaseNotification]的 Pull-Point 接口的实施是可选的。

一个 ONVIF 兼容设备应该安装 Base Subscription Manager Interface 更新和退订业务。这个可终止的订阅界面是可选的，就像 WS-Resources 是可选的那样。

15.2 实时拉点通知接口

本节介绍了实时拉取点通知接口. 这个接口提供防火墙友好通知接口实时轮询和启动所有客户端通讯.

该接口用以下方法进行使用：

1：客户端要求用 CreatePullPointSubscriptionRequest 消息为设备配置 PullPointSubscription。

2：当订阅被接受时，设备评估订阅要么返回一个 createpullpointsubscriptionresponse 信号，要么返回所有误码中的一个误码。

3：如果订阅被接受，这个应该包含一个 WS-EndpointReference 对一个 SubscriptionManager。这个 WS-Endpoint 应该提供一个 PullMessages 操作（使用客户端检索通知并且订阅管理界面的接口被描述在规范[WS-BaseNotification]中）。Base Subscription Manager 接口组成了 PullMessages 更新和退订业务操作。序列图的相互作用被显示在图 22. 该 PullMessagesRequest 包含超时和信息限制参数。

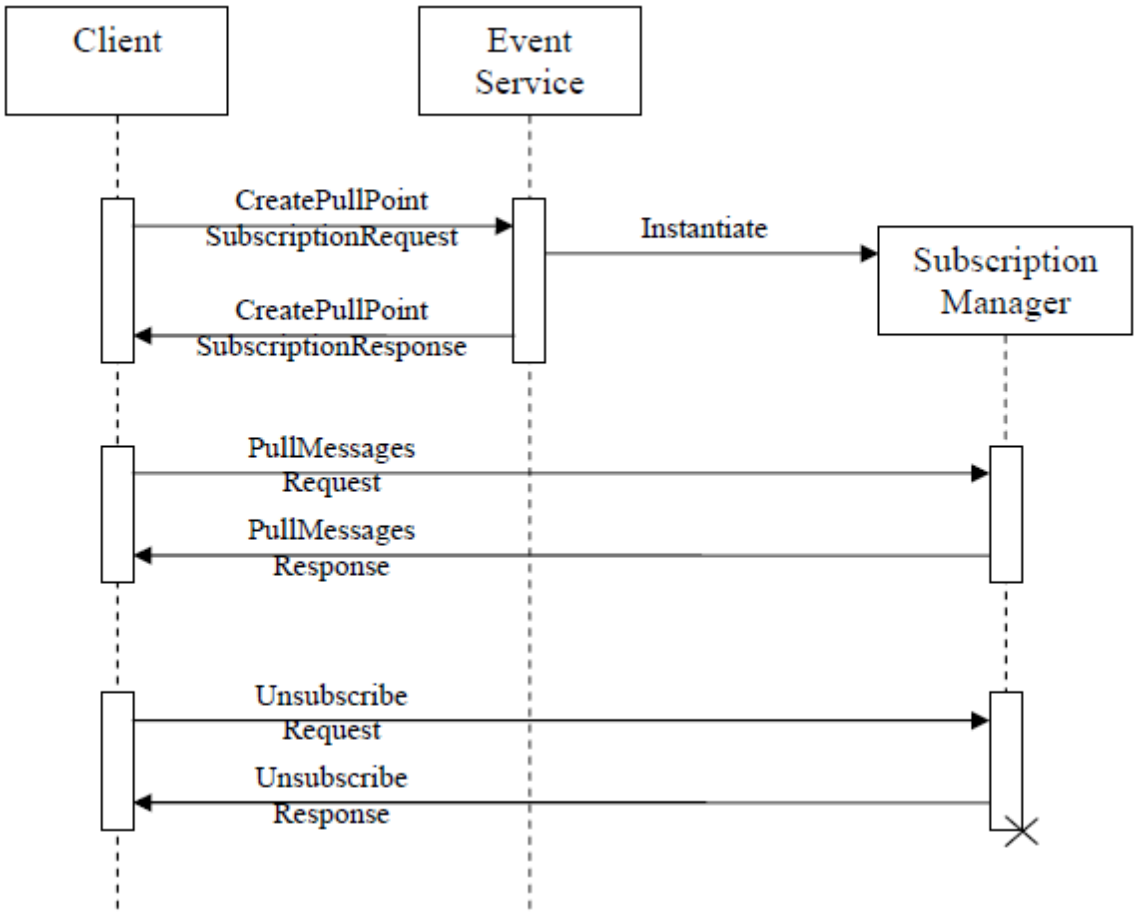


图 22：实时通知接口的序列图

4：该设备应该立即用通知回应已经汇总的客户代表。如果没有聚集的通知，这个设备将用它自己的应答等待，直到要么产生一个客户端通知，要么指定的时间超时了。在一些情况下，将包含一些应答，具体的通知数量由消息限制参数决定。每个 PullMessagesResponse 应答之后，客户端可以实时查询的通知当启动一个新的 PullMessagesRequest

5：如果既不是终止的时间也不是相对终止的时间，在 CreatePullPointSubscriptionRequest 设置中，每个 PullMessagesRequest 可能被中断，作为一个与 keep-alive 对相应的

PullPointSubscription。通过相对的终止时间，如果它是可得到的话或者根据设备的内部默认值，那么终止时间将被重新计算。为了通知客户端的更新时间，PullMessagesReponse 应该包含当前时间和终止时间要素。当 PullMessagesRequest 被用作 keep-alive 为相应的 PullPointSubscriptio，更新请求，在[WS-BaseNotification]中定义的，客户端不必呼叫，不过该设备应支持它的 PullPointSubscription。

15.2.1 创建 pull point subscription

设备应该提供如下的创建 PullPointSubscription 命令

表 210：CreatePullPointSubscription 命令

CreatePullPointSubscription	请求-应答
信息名字	描述
CreatePullPointSubscription 要求	作为 [WS-BaseNotification]的 SubscriptionRequest 这一消息包含相同的元素 wsnt:FilterType Filter [0][1] wsnt:AbsoluteOrRelativeTimeType InitialTerminationTime [0][1] xs:any SubscriptionPolicy [0][1]
CreatePullPointSubscription 应答	作为 [WS-BaseNotification]的 SubscriptionRequest 这一应答包含相同的元素
误码	描述
	作为相同的 Subscription Request 结果被使用。

15.2.2 pull 消息

设备应提供以下的 PullMessages 命令，对于所有的 SubscriptionManager 端点（被 CreatePullPointSubscription 命令返回的）

表 211：PullMessages command

PullMessages	请求-应答
信息名字	描述
PullMessages 要求	此信息应处理一个 SubscriptionManager 得信息为了 pull 通知 xs:duration Timeout [1][1] xs:int MessageLimit [1][1]
PullMessages 应答	对 SubscriptionManager 应包含相应一个名单的通知（连接一个更新的终止时间）
PullMessages 的错误应答	要么终止时间要么信息限制超过上限应该被设备支持，故障信息应当包括上限参数 xs:duration MaxTimeout[1][1]

	xs:int MaxMessageLimit[1][1]
误码	描述
	没有指定的误码

15.3 通知流接口

11.10 节介绍了建立，删除和元数据的修改配置。某些元数据的配置可能包含多个结构相同的订阅。当一个数据的配置已被分配到一个文件的订阅中，客户端将使用一个简介，得到一个作为元数据流插入的包括配置通知。凭借 RTP,实时传输协议流应该采取一个合发布通过一个 ONVIF 兼容设备。

[WS-BaseNotification]定义了一个要素 wsnt:NotificationMessage 来包装有效的消息载荷，主题和 ProducerReference。这个消息的结构与直接的通知请求相同。多元化的要素可以被放置在一个元数据文件（这部分的介绍在实时查看章节）
没有明确的流通知的 SubscriptionReference，因此本不能含有 SubscriptionReference 元素。

15.4 属性

一个属性是一个名字的集合，数值对代表了一个独特的可设置数据。通过它们结合的就像普通事件一样的主题，源，关键值和包装，它们是唯一确定的。属性还包括一个额外的标志用以说明它是否是新建的，改变了吗，还是被删除了。
当一个客户端订阅了一个代表着一定价值的属性，该设备应提供一些通知，来通知客户端的所有对象的请求属性。客户也可以要求积极的属性的数值，客户端在任何时间的订阅通过要求一个同步点。
为了将所有与事情相关的属性进行分组并且呈现出目前一致的客户，就在此标准中明确了特定的属性接口。只要适用，都推荐使用此属性接口。章节 15.5 详细的解释了事件和属性的结构。

15.4.1 属性举例

下面的视频分析实例显示了动态的属性行为：本规则的引擎接口的视频分析探测器可以定义域。这种探测器域用一个多边形的图像平面来描述。对于场景中的每一个对象，引擎规则确定哪些对象是在多边形内。通过订阅相应的探测器领域 ObjectsInside 属性，客户端可以获得这些信息。每次一个对象出现在现场，一个新的 ObjectsInside 属性就被创建了。客户就通过一个相关的属性创建通知（暗示是否对象出现在里面或者多边形的外面）告知了。每一个对象进入或者离开多边形，将产生一个属性改变的通知，用以显示 ObjectsInside 属性来告诉这个对象已经改变了，当一个对象离开了这个场景，相应的 ObjectsInside
属性就被删除了，并且通过一个属性删除通知来告诉客户。

15.5 通知结构

下面的代码模式是针对 wsnt:NotificationMessage [WSBaseNotification]:

```
<xs:complexType name="NotificationMessageHolderType" >
<xs:sequence>
<xs:element ref="wsnt:SubscriptionReference" minOccurs="0" />
<xs:element ref="wsnt:Topic" minOccurs="0" />
<xs:element ref="wsnt:ProducerReference" minOccurs="0" />
<xs:element name="Message">
<xs:complexType>
<xs:sequence>
<xs:any namespace="##any" processContents="lax" />
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
<xs:element name="NotificationMessage"
type="wsnt:NotificationMessageHolderType"/>
```

这些与下面的 XML structure 相关：

```
<wsnt:NotificationMessage>
<wsnt:SubscriptionReference>
wsa:EndpointReferenceType
</wsnt:SubscriptionReference>
<wsnt:Topic Dialect="xs:anyURI">
…
</wsnt:Topic>?
<wsnt:ProducerReference>
wsa:EndpointReferenceType
</wsnt:ProducerReference>
<wsnt:Message>
…
</wsnt:Message>
</wsnt:NotificationMessage
```

wsnt:Message 地方的要素包含了实际有效载荷的通知。消息要素的 XML 数据类型可以指定在一个 TopicTree 定义中（看 15.7 章节）

15.5.1 章节给出了一个客户端检索信息的概述通过通知。

15.5.2 章节给出了详细的有效载荷消息的格式。

15.5.4 章节介绍了一种描述语言得信息。

15.5.5 定义了使用在订阅的过滤器语法中的语法通过他们的消息内容。

15.5.1 通知消息

通知至少下列问题的答案

什么时候发生？

谁触发了这个事情？

发生了什么？

“什么时候”这个问题靠添加一个时间属性的信息元素通知来给出了答案。一个 ONVIF 兼容设备应该包括时间属性的信息元。

“谁”这个问题被分为两部分。一部分是 WS-Endpoint（标识设备或产生这个通知的设备之内的服务)因此，这个 WS-Endpoint 应该指定在 NotificationMessage 的 ProducerReference Element 之内。其次是确定的组成部分在 WS-Endpoint（负责通知的产生）之内。取决于多个组件的参数或者一个都不， 可能需要确定组件的唯一性。这些参数是作为项目被放置在元素的信息容器内的。

对“发生了什么”这个问题用两步来回答。首先通知信息的标志元素被用来分类事件。接着为了描述详细的事情，项目被添加到数据元素的信息容器中。

当说到主题点得属性（看章节 15.4），客户使用通知发生器，这个话题，源项目和可选的关键项目，为了确认这个属性，这些值应该对唯一的标示符起作用。

15.5.1.1 事件例子

随后的例子表明，不同地区的通知：

```
<wsnt:NotificationMessage>
...
<wsnt:Topic Dialect="...Concrete">
tns1:PTZController/PTZPreset/Reached
</wsnt:Topic>
<wsnt:Message>
<tt:Message UtcTime="...">
<tt:Source>
<tt:SimpleItem Name="PTZConfigurationToken" Value="PTZConfig1"/>
</tt:Source>
<tt:Data>
<tt:SimpleItem Name="PresetToken" Value="Preset5"/>
<tt:SimpleItem Name="PresetName" Value="ParkingLot"/>
</tt:Data>
</tt:Message>
</wsnt:Message>
</wsnt:NotificationMessage>
```

“PTZConfigurationToken”项目确定了独特的组件（负责事件的检测），在这个例子中，这个组件是一个 PTZ 节点（被 PTZ 配置 “PTZConfig1”引用的）. 事件 tns1:PTZController/PTZPreset/Reached 暗示了 PTZ 单元到了一个预设的装置。数据块包含了拿个预设的装置是它的信息。从而，预设由一个名字为“PresetName”的 PresetToken “Preset5 确定了下来。

15.5.2 消息格式

通知消息的消息元素被定义在[ONVIF Schema].中

这个定义在下面展示出来：

```
<xs:element name="Message" type="Message">
    <xs:element name="Message">
    <xs:complexType>
```

```
<xs:sequence>
<xs:element name="Source" type="tt:ItemList" minOccurs="0"/>
<xs:element name="Key" type="tt:ItemList" minOccurs="0"/>
<xs:element name="Data" type="tt:ItemList" minOccurs="0"/>
...
</xs:sequence>
<xs:attribute name="UtcTime" type="xs:time" use="required"/>
<xs:attribute name="PropertyOperation" type="tt:PropertyOperationType"/>
</xs:complexType>
</xs:element>
<xs:complexType name="ItemList">
<xs:sequence>
<xs:element name="SimpleItem" minOccurs="0" maxOccurs="unbounded">
<xs:complexType>
<xs:attribute name="Name" type="xs:string" use="required"/>
<xs:attribute name="Value" type="xs:anySimpleType" use="required"/>
</xs:complexType>
</xs:element>
<xs:element name="ElementItem" minOccurs="0" maxOccurs="unbounded">
<xs:complexType>
<xs:sequence>
<xs:any namespace="##any"/>
</xs:sequence>
<xs:attribute name="Name" type="xs:string" use="required"/>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
<xs:simpleType name="PropertyOperationType">
<xs:restriction base="xs:string">
<xs:enumeration value="Initialized"/>
<xs:enumeration value="Deleted"/>
<xs:enumeration value="Changed"/>
</xs:restriction>
</xs:simpleType>
```

在消息元素范围之内的项目被分为了三类：来源，关键点，和数据。关键点这一组不应该被与属性不相关的通知使用。

多个简单的元素和项目可以被分在不同的组里面，每一组有一个名字和值，在一个 ElementItem 里面，该值被表示为了一个 XML 元素。在 SimpleItem 里面，该值被价值属性指定。该项目的名字必须在所有被包含的项目中是唯一的

一般推荐用 SimpleItem 代替 ElementItem，无论什么时候可用。自从 SimpleItems 轻松的整合到一个通用的客户端的时候。确切的既简单又 ElementItem 类型信息可以从 topicset（每一个消息可以增加一个描述消息负载的地方）中提取。

当属性相关的通知产生时，属性操作应该被显示。“Initialized”操作码应该用来通知客

户关于这个属性的创建。
“Initialized”操作码应该被使用当同步点被请求的时候

15.5.3 属性举例，持续

在章节 15.4.1 中的例子被要求了一个关键项目的选项，本节中的示例显示了关键项目的应用。

引擎规则可能包含了 FieldDetector 规则。为每一个现场的对象，这些规则定义了一个 ObjectsInside 属性。当新对象出现在这样的领域外，下面的通知就产生了：

```
<wsnt:NotificationMessage>
...
<wsnt:Topic Dialect="...Concrete">
tns1:RuleEngine/FieldDetector/ObjectsInside
</wsnt:Topic>
<wsnt:Message>
<tt:Message UtcTime="..." PropertyOperation="Initialized">
<tt:Source>
<tt:SimpleItem Name="VideoSourceConfigurationToken" Value="1"/>
<tt:SimpleItem Name="VideoAnalyticsConfigurationToken" Value="1"/>
<tt:SimpleItem Name="Rule" Value="myImportantField"/>
</tt:Source>
<tt:Key>
<tt:SimpleItem Name="ObjectId" Value="5"/>
</tt:Key>
<tt:Data>
<tt:SimpleItem Name="IsInside" Value="false"/>
</tt:Data>
</tt:Message>
</wsnt:Message>
</wsnt:NotificationMessage>
```

源项目描述了产生这个项目的规则，当多个对象出现在现场的时候，每一个对象都拥有一个它自己所有的 ObjectsInside 属性，因此这些对象的身份被作为一个额外得源项目来使用，为了使属性的唯一性。这个 IsInside 项目是一个布尔值，指示对象是领域的内部还是外部。
当对象进入一个领域，产生一个属性改变信息的规则类似于以下：

```
<wsnt:NotificationMessage>
...
<wsnt:Topic Dialect="...Concrete">
tns1:RuleEngine/FieldDetector/ObjectsInside
</wsnt:Topic>
<wsnt:Message>
<tt:Message UtcTime="..." PropertyOperation="Changed">
<tt:Source>
<tt:SimpleItem Name="VideoSourceConfigurationToken" Value="1"/>
<tt:SimpleItem Name="VideoAnalyticsConfigurationToken" Value="1"/>
<tt:SimpleItem Name="Rule" Value="myImportantField"/>
```

```
</tt:Source>
<tt:Key>
<tt:SimpleItem Name="ObjectId" Value="5"/>
</tt:Key>
<tt:Data>
<tt:SimpleItem Name="IsInside" Value="true"/>
</tt:Data>
</tt:Message>
</wsnt:Message>
</wsnt:NotificationMessage>
```

最后，当对象离开现场的时候，一个属性删除信息就产生了：

```
<wsnt:NotificationMessage>
...
<wsnt:Topic Dialect="...Concrete">
tns1:RuleEngine/FieldDetector/ObjectsInside
</wsnt:Topic>
<wsnt:Message>
<tt:Message UtcTime="..." PropertyOperation="Deleted">
<tt:Source>
<tt:SimpleItem Name="VideoSourceConfigurationToken" Value="1"/>
<tt:SimpleItem Name="VideoAnalyticsConfigurationToken" Value="1"/>
<tt:SimpleItem Name="Rule" Value="myImportantField"/>
</tt:Source>
<tt:Key>
<tt:SimpleItem Name="ObjectId" Value="5"/>
</tt:Key>
</tt:Message>
</wsnt:Message>
</wsnt:NotificationMessage
```

15.5.4 信息描述语言

有效载荷的信息结构是在前一节介绍的，这个结构包含了三个分组：源，重点和数据。每组都包含一套 Simple 和 ElementItems。对于每个主题来说，一个设备能够描述这样的项目（产生通知将是其中的一部分，通过使用这个信息描述语言）。下面的描述语言描述了强制性的消息项目 6：

```
<xs:complexType name="MessageDescription">
<xs:sequence>
<xs:element name="Source" type="tt:ItemListDescription"
minOccurs="0"/>
<xs:element name="Key" type="tt:ItemListDescription" minOccurs="0"/>
<xs:element name="Data" type="tt:ItemListDescription" minOccurs="0"/>
...
</xs:sequence>
<xs:attribute name="IsProperty" type="xs:boolean"/>
```

```
</xs:complexType>
<xs:complexType name="ItemListDescription">
<xs:sequence>
<xs:element name="SimpleItemDescription"
minOccurs="0" maxOccurs="unbounded">
<xs:complexType>
<xs:attribute name="Name" type="xs:string" use="required"/>
<xs:attribute name="Type" type="xs:QName" use="required"/>
</xs:complexType>
</xs:element>
<xs:element name="ElementItemDescription"
minOccurs="0" maxOccurs="unbounded">
<xs:complexType>
<xs:attribute name="Name" type="xs:string" use="required"/>
<xs:attribute name="Type" type="xs:QName" use="required"/>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
```

在所有的独立项目之内，项目的名字属性将是唯一的。当描述的消息涉及到财产的时候，该 IsProperty 属性应该被设置成真。然而，这个消息不涉及到一个财产，重点组将不被展现出来。SimpleItemDescriptor 的类型属性应该匹配这个 XML 格式定义的 SimpleElement。相似的，一个 ElementItemDescriptor 的类型属性也应该匹配 XML 格式的目标元素的声明。

过去常常描述有效消息载荷的本地格式文件被列举在 GetEventPropertiesResponse 消息中（章节 15.8）

15.5.4.1 消息描述举例

下面的代码是一个消息描述的例子，相关财产的描述例子在 15.5.3

```
<tt:MessageDescription IsProperty="true">
<tt:Source>
<tt:SimpleItemDescriptionDescription Name="VideoSourceConfigurationToken"
Type="tt:ReferenceToken"/>
<tt:SimpleItemDescriptionDescription Name="VideoAnalyticsConfigurationToken"
Type="tt:ReferenceToken"/>
<tt:SimpleItemDescriptionDescription Name="Rule"
Type="xs:string"/>
</tt:Source>
<tt:Key>
<tt:SimpleItemDescriptionDescription Name="ObjectId"
Type="tt:ObjectRefType"/>
</tt:Key>
<tt:Data>
<tt:SimpleItemDescriptionDescription Name="IsInside"
Type="xs:boolean"/>
</tt:Data>
```

</tt:MessageDescription>

15.5.5 消息内容过滤器

在请求订阅中，靠 TopicExpression (see Section15.7.3) and by MessageContent ，客户端可以过滤通知。对于后者来说[WS-BaseNotification]提出了一个 XPath1.0 的特定的语言。在此规范中，由于要求了特定的信息结构，这个规范要求 XPath 1.0 的句法的一个子集。一个 ONVIF 的兼容设备应该安装 XPath 1.0 的一个子集。相关的特定语言可以参考 URI: Dialect=http://www.onvif.org/ver10/tev/messageContentFilter/ItemFilter

优先级与结合性：

这个“与”操作比“或”操作有更高的优先级，“与”和“或”都是左结合的。

在以下的语法定义中，“与”和“或”操作的优先级和结合性与 XPath 1.0 中的是相同的。

表达式的结构如下：

[1] Expression ::= BoolExpr | Expression ‘and’ Expression

| Expression ‘or’ Expression | ‘(‘ Expression ‘)’ | ‘not’ ‘(‘ Expression ‘)’

[2] BoolExpr ::= ‘boolean’ ‘(‘ PathExpr ‘)’

[3] PathExpr ::= [‘//’Prefix?’SimpleItem‘ | ’//’Prefix?’ElementItem‘] NodeTest

[4] Prefix::= NamespacePrefix’:’ | ‘’

[5] NodeTest ::= ‘[‘ AttrExpr ‘]’

[6] AttrExpr ::= AttrComp | AttrExpr ‘and’ AttrExpr | AttrExpr ‘or‘ AttrExpr | ‘(‘ AttrExpr ‘)‘

| ‘not’ ‘(‘ AttrExpr ‘)‘

[7] AttrComp ::= Attribute ‘=’ ‘””’ String ‘””’

[8] Attribute ::= ‘@Name’ | ‘@Value’

这个语法允许测试的存在，在(Source, Key or Data).的组中独立的 Simple or ElementItems。此外，SimpleItems 的值可能被检查。The SimpleItem and ElementItem Prefix 名字空间可能与“http://www.onvif.org/ver10/schema.相关。

最后，这些测试可能是的任意布尔组合。可以指定下面的表达式：

只返回包含视频源配置 1 的参考的通知。

boolean(//tt:SimpleItem[@Name=”VideoSourceConfigurationToken” and @Value=”1”])

返回不包含视频分析配置的参考的通知。

not(boolean(//tt:SimpleItem[@Name=”VideoAnalyticsConfigurationToken”]))

返回与正运行在视频原配置 1 上的视频分析配置 2 确切关联的的通知。

boolean(//tt:SimpleItem[@Name=”VideoAnalyticsConfigurationToken” and @Value=”2”])
and
boolean(//tt:SimpleItem[@Name=”VideoSourceConfigurationToken” and @Value=”1”])

返回与视频原配置相关联但与视频分析配置不关联的通知。

boolean(//tt:SimpleItem[@Name=”VideoSourceConfigurationToken” and @Value=”1”])
and
not(boolean(//tt:SimpleItem[@Name=”VideoAnalyticsConfigurationToken”]))

返回通知，当对象进入或者出现在““myImportantField”。

boolean(//tt:SimpleItem[@Name=”IsInside” and @Value=”true”])
and
boolean(//tt:SimpleItem[@Name=”Rule” and @Value=”myImportantField”])

15.6 同步点

在 15.4 章节介绍的性能，通知客户用一个统一的方式对属性创建，改变和删除。当一个客户想与设备属性一起参与性能的设置，它可以请求重复所有目前状态属性的同步点。所有产生了通知的属性操作被设置到了“Initialized”（看章节 15.5）。同步点从要么 SubscriptionResponse 返回 要么在 CreatePullPointSubscriptionResponse 返回。通过通知接口的通知运输，更新的属性就被传递出去了。通过所有的订阅管理终端，提供了以下的操作：

表 212：设置同步命令点命令

设置同步命令点	请求-应答
消息名字	描述
设置同步命令点请求	消息空
设置同步命令点应答	消息空
误码	描述
	没有指定的命令错误

当客户使用一个流接口通知时，客户应该使用设置同步点的操作（被定义在媒体服务，看章节 11.18）

15.7 主题结构

本标准扩展的主题框架在[WS-Topics]规范中定义了。章节 15.7.1 描述了一个 ONVIF 主题名字空间（对于供应商的具体议题，这应该被视为最基本的）。附录 0 为这些扩展列举了典型的例子，章节 15.7.2 定义了一个主题属性的接口。这些界面应该被一个 ONVIF 兼容设备采取。章节 15.7.3 包含了信息描述语言。所有的主题都从 ONVIF 主题名字空间描述主题类型通过 15.7.3 章节。章节 15.7.3 定于了设备支持的主题表达方言（Topic Expression Dialects）。

15.7.1 ONVIF主题名字空间

[WS-Topics]规范区别某一个主题命名空间的主题树与一个网站服务设置的主题。这个区别允许供应商提及一个共同话题的命名空间，当只使用该部分的主题的时候。
如果一个存在名字空间的主题树包括了一个可用主题的一个子集，通过定义一个新的主题名字空间，种植这个主题树。一个新的主题名字空间就确定了，通过对存在的一个主题名字空间追加一个新的话题，就像[WS-Topics]规范描述的那样。

以下的根话题定义在 ONVIF 名字空间中，所有提到这些话题的通知应该使用消息格式就像章节 15.5.2 描述的那样：

```
<wstop:TopicNamespace name="ONVIF"
targetNamespace="http://www.onvif.org/ver10/topics" >
<wstop:Topic name="Device"/>
<wstop:Topic name="VideoSource"/>
<wstop:Topic name="VideoEncoder"/>
<wstop:Topic name="VideoAnalytics"/>
<wstop:Topic name="RuleEngine"/>
<wstop:Topic name="PTZController"/>
<wstop:Topic name="AudioSource"/>
<wstop:Topic name="AudioEncoder"/>
<wstop:Topic name="UserAlarm"/>
<wstop:Topic name="MediaControl"/>
<wstop:Topic name="RecordingConfig"/>
<wstop:Topic name="RecordingHistory"/>
<wstop:Topic name="VideoOutput"/>
<wstop:Topic name="AudioOutput"/>
<wstop:Topic name="VideoDecoder"/>
<wstop:Topic name="AudioDecoder"/>
<wstop:Topic name=”Receiver”/>
</wstop:TopicNamespace>
```

15.7.2 主题类型信息

类型信息被添加到一个主题元素的下面，通过增加一个定义在 15.5.4 中的消息描述类型的消息描述元素。

主题元素可以被 wstop:topic attribute 用 "true"值来确定.

下面的例子显示了一个主题设置的主题如何与消息描述被讨论：

```
<tns1:RuleEngine wstop:topic="true">
<tns1:LineDetector wstop:topic="true">
<tns1:Crossed wstop:topic="true">
<tt:MessageDescription>
<tt:Source>
<tt:SimpleItemDescription Name="VideoSourceConfigurationToken"
Type="tt:ReferenceToken"/>
<tt:SimpleItemDescription Name="VideoAnalyticsConfigurationToken"
Type="tt:ReferenceToken"/>
<tt:SimpleItemDescription Name="Rule" Type="xs:string"/>
</tt:Source>
<tt:Data>
<tt:SimpleItemDescription Name="ObjectId" Type="tt:ObjectRefType"/>
</tt:Data>
</tt:MessageDescription>
```

```
</tns1:Crossed>
</tns1:LineDetector>
<tns1:FieldDetector wstop:topic="true">
<tns1:ObjectsInside wstop:topic="true">
<tt:MessageDescription IsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="VideoSourceConfigurationToken"
Type="tt:ReferenceToken"/>
<tt:SimpleItemDescription Name="VideoAnalyticsConfigurationToken"
Type="tt:ReferenceToken"/>
<tt:SimpleItemDescription Name="Rule" Type="xs:string"/>
</tt:Source>
<tt:Key>
<tt:SimpleItemDescription Name="ObjectId" Type="tt:ObjectRefType"/>
</tt;Key>
<tt:Data>
<tt:SimpleItemDescription Name="IsInside" Type="xs:boolean"/>
</tt:Data>
</tt:MessageDescription>
</tns1:ObjectsInside>
</tns1:FieldDetector>
</tns1:RuleEngine>
```

15.7.3 主题过滤器

一个 ONVIF 兼容设备应该支持一个定义在[WSTopics]规范中的具体主题表达。在主题范围树内这个规范定义了一个指定的主题范围。下面的话应该被确定，当一个具体的主题表达在订阅的过滤器中作为一个 TopicExpression 来使用。

http://docs.oasis-open.org/wsn/t-1/TopicExpression/Concrete

以下的主题表达格式应该被设备支持。

通过使用一个“或”操作，加上一个与话题子树匹配的字符串，语法可以扩展具体的课题表达式。这个扩展的语法允许一个任意主题设置的选择在一个订阅之类。

作为主题表达语法被相同的方法描述在[WS-Topics 1.3]规范中

[3] TopicExpression ::= TopicPath (‘|’ TopicPath)*

[4] TopicPath ::= RootTopic ChildTopicExpression* ('//.')?

[5] RootTopic ::= QName

如果一个命名空间的前缀包括在 RootTopic 之内，它应该符合一个有效的话题命名空间定义和本地名称对应的一个根主题的名称被定义在这个命名空间之中。

[6] ChildTopicExpression ::= ‘/’ ChildTopicName

[7] ChildTopicName ::= QName | NCName

这个 NCname 和这个 QNAME 的本地部分应该与主题的名称对应起来在后续路径之内（在每一个斜线表示另一个子主题元素的层次在这个路径中。）

为了参考这个 TopicExpression 特定语言，下面的 URI 应该被使用：

Dialect=http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet

如果 TopicExpression 的结尾用了一个“//”字符，它暗示这个 TopicExpression 匹配一个话题

子树。比如：

“tns1:RuleEngine/FieldDetector//.”

这个确定了子树组成 tns1：引擎规则/领域探测器和它所有的后裔

下面的示例演示了 ConcreteSet 主题表达的用法：

搜寻通知（有视频分析主题的）作为双亲主题：

```
<wsnt:TopicExpression Dialect =
        "http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet" >
        tns1:VideoAnalytics//.
        </wsnt:TopicExpression>
```

搜寻通知（有视频分析和引擎规则的）作为双亲主题：

```
<wsnt:TopicExpression Dialect =
        "http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet" >
        tns1:VideoAnalytics//.|tns1:RuleEngine//.
        </wsnt:TopicExpression>
```

搜寻由线路探测器或者领域探测器产生的通知：

```
<wsnt:TopicExpression Dialect =
        "http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet">
        tns1:RuleEngine/FieldDetector//.|tns1:RuleEngine/LineDetector//.
        </wsnt:TopicExpression>
```

15.8 获取事件属性

[WS-BaseNotification]规则定义了一个 WS-Resouce 选项属性。这个规范不要求 WS-ResourceProperty 接口来执行。相反的，接下来的直接接口应该靠一个 ONVIF 兼容设备来直接执行，为了提供一个设备支持的过滤特定语言，文件格式和主题的信息。

表 213：获取时间的属性命令

获取时间属性	请求-应答
信息名字	描述
获取时事件性的请求	这是一个空消息
获取事件属性应答	xs:anyURI TopicNamespaceLocation [1][unbounded] xs:boolean FixedTopicSet [1][1] wstop:TopicSetType TopicSet [1][1] xs:anyURI TopicExpressionDialect [1][unbounded] xs:anyURI MessageContentFilterDialect [1][unbounded] xs:anyURI ProducerPropertiesFilterDialect [0][unbounded] xs:anyURI MessageContentSchemaLocation [1][unbounded]

误码	描述
	没有命令指定错误

一个 ONVIF 兼容设备应该做出回应，并且声明它的 TopicSet 是否是固定的，哪些主题是提供了的，支持哪些方言。

接下来的主题表达语言是 ONVIF 兼容设备指令的(看 15.7.3 章节)：

http://docs.oasis-open.org/wsn/t-1/TopicExpression/Concrete

http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet

下面的 MessageContentFilterDialects 是 ONVIF 兼容设备指令的（看 15.5.5 章节）

http://www.onvif.org/ver10/tev/messageContentFilter/ItemFilter

这个设备并不要求任何 ProducerPropertiesDialect 的支持

消息内容描述语言，在 15.5.4 介绍的允许参考指定的类型。为了缓解这些类型整合到客户端应用程序

GetEventPropertiesResponse 应该列出所有的 URI 方位来架构那些型号已经被用到了通知的描述中的文件

带有 MessageContentSchemaLocation 要素。这表至少包含 ONVIF 格式文件的 URI

15.9 SOAP 错误消息

当处理要么来自客户要么来自 Subscription Manager 的消息时，如果设备遇到失败，设备将产生一个 SOAP 1.2 的错误消息。

所有的 SOAP1.2 错误消息应该产生根据[WS-BaseNotification] 和 [WSTopics]规范

15.10 通知例子

下面的例子是一个完整的通信模式的通知。它采用实时的通知接口 Pull-Point 来接收通知的。

15.10.1 获取事件属性请求

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
xmlns:SOAP-ENV=http://www.w3.org/2003/05/soap-envelope
xmlns:wsa="http://www.w3.org/2005/08/addressing"
xmlns:tet="http://www.onvif.org/ver10/events/wsdl">
<SOAP-ENV:Header>
<wsa:Action>
http://www.onvif.org/ver10/events/wsdl/EventPortType/GetEventPropertiesRequest
</wsa:Action>
</SOAP-ENV:Header>
<SOAP-ENV:Body>
<tet:GetEventProperties>
</tet:GetEventProperties>
```

```
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

15.10.2 获取事件属性应答

在这个例子中，设备应答使用了 ONVIF 主题名字空间(这个描述可以从 http://www.onvif.org/onvif/ver10/topics/topicns.xml 下载到)这个主题设置不随时间的变化并且组成一个单一的主题 tns1:RuleEngine/LineDetector/Crossed.与这个主题联系在一起的消息包含关于 VideoSourceConfigurationToken 的信息，VideoAnalyticsConfigurationToken 和已经过线的对象。设备支持下面两种 TopicExpressionDialects：

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
xmlns:wsa="http://www.w3.org/2005/08/addressing"
xmlns:wstop="http://docs.oasis-open.org/wsn/t-1"
xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2"
xmlns:tet="http://www.onvif.org/ver10/events/wsdl"
xmlns:tns1="http://www.onvif.org/ver10/topics"
xmlns:tt="http://www.onvif.org/ver10/schema">
<SOAP-ENV:Header>
<wsa:Action>
http://www.onvif.org/ver10/events/wsdl/EventPortType/GetEventPropertiesResponse
</wsa:Action>
</SOAP-ENV:Header>
<SOAP-ENV:Body>
<tet:GetEventPropertiesResponse>
<tet:TopicNamespaceLocation>
http://www.onvif.org/onvif/ver10/topics/topicns.xml
</tet:TopicNamespaceLocation>
<wsnt:FixedTopicSet>
true
</wsnt:FixedTopicSet>
<wstop:TopicSet>
<tns1:RuleEngine>
<tns1:LineDetector>
<tns1:Crossed wstop:topic="true">
<tt:MessageDescription>
<tt:Source>
<tt:SimpleItemDescription Name="VideoSourceConfigurationToken"
Type="tt:ReferenceToken"/>
<tt:SimpleItemDescription Name="VideoAnalyticsConfigurationToken"
Type="tt:ReferenceToken"/>
</tt:Source>
<tt:Data>
```

<tt:SimpleItemDescription Name="ObjectId"
Type="tt:ObjectRefType"/>
</tt:Data>
</tt:MessageDescription>
</tns1:Crossed>
</tns1:LineDetector>
</tns1:RuleEngine>
</wstop:TopicSet>
<wsnt:TopicExpressionDialect>
http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet
</wsnt:TopicExpressionDialect>
<wsnt:TopicExpressionDialect>
http://docs.oasis-open.org/wsnt/t-1/TopicExpression/ConcreteSet
</wsnt:TopicExpressionDialect>
<wsnt:MessageContentFilterDialect>
http://www.onvif.org/ver10/tev/messageContentFilter/ItemFilter
</wsnt:MessageContentFilterDialect>
<tt:MessageContentSchemaLocation>
http://www.onvif.org/onvif/ver10/schema/onvif.xsd
</tt:MessageContentSchemaLocation>
</tet:GetEventPropertiesResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

15.10.3 创建PULLPOIT订阅

客户可以用 TopicProperties 信息订阅具体的通知。下面的 XML 例子为设备引擎规则产生的通知显示订阅。客户仅仅通过参考 VideoAnalyticsConfiguration “2” and VideoSourceConfiguration “1”.的通知给出反应。这个订阅有一分钟的超时，如果订阅没有明确更新的或者消息没有定期的拉出来，那么一分钟之后将自动结束。

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
xmlns:wsa="http://www.w3.org/2005/08/addressing"
xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2"
xmlns:tet="http://www.onvif.org/ver10/events/wsdl"
xmlns:tns1="http://www.onvif.org/ver10/topics">
<SOAP-ENV:Header>
<wsa:Action>
http://www.onvif.org/ver10/events/wsdl/EventPortType/CreatePullPointSubscriptionR
eq
uest
</wsa:Action>
</SOAP-ENV:Header>
<SOAP-ENV:Body>

```
<tet:CreatePullPointSubscription>
<tet:Filter>
<wsnt:TopicExpression
Dialect="http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet">
tns1:RuleEngine//.
</wsnt:TopicExpression>
<wsnt:MessageContent
Dialect="http://www.onvif.org/ver10/tev/messageContentFilter/ItemFilter">
boolean(//tt:SimpleItem[@Name="VideoAnalyticsConfigurationToken"
and @Value="2"] ) and
boolean(//tt:SimpleItem[@Name="VideoSourceConfigurationToken"
and @Value="1"] )
</wsnt:MessageContent>
</tet:Filter>
<tet:InitialTerminationTime>
PT1M
</tet:InitialTerminationTime>
</tet:CreatePullPointSubscription>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

15.10.4 创建PULLPOIT订阅应答

当设备接受订阅的时候，它将返回 http://160.10.64.10/Subscription?Idx=0 代表了订阅终点的 URI。此外，将通知客户关于当前设备的时间和创建订阅的终止时间。

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
xmlns:SOAP-ENV=http://www.w3.org/2003/05/soap-envelope
xmlns:wsa="http://www.w3.org/2005/08/addressing"
xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2"
xmlns:tet="http://www.onvif.org/ver10/events/wsdl">
<SOAP-ENV:Header>
<wsa:Action>
http://www.onvif.org/ver10/events/wsdl/EventPortType/CreatePullPointSubscription
Response
</wsa:Action>
</SOAP-ENV:Header>
<SOAP-ENV:Body>
<tet:CreatePullPointSubscriptionResponse>
<tet:SubscriptionReference>
<wsa:Address>
http://160.10.64.10/Subscription?Idx=0
</wsa:Address>
</tet:SubscriptionReference>
<wsnt:CurrentTime>
```

```
2008-10-09T13:52:59
</wsnt:CurrentTime>
<wsnt:TerminationTime>
2008-10-09T13:53:59
</wsnt:TerminationTime>
</tet:CreatePullPointSubscriptionResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

15.10.5 拉消息请求

客户发送一个拉取信息的请求给终点（在 CreatePullPointSubscriptionResponse 中给出的终点）目的是获得一个与订阅相关的通知。下面的样本要求包含一个 5 秒的超时，并且限制这个应答消息的总数量到 2.

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
xmlns:wsa="http://www.w3.org/2005/08/addressing"
xmlns:tet="http://www.onvif.org/ver10/events/wsdl" >
<SOAP-ENV:Header>
<wsa:Action>
http://www.onvif.org/ver10/events/wsdl/PullPointSubscription/PullMessagesRequest
</wsa:Action>
<wsa:To>http://160.10.64.10/Subscription?Idx=0</wsa:To>
</SOAP-ENV:Header>
<SOAP-ENV:Body>
<tet:PullMessages>
<tet:Timeout>
PT5S
</tet:Timeout>
<tet:MessageLimit>
2
</tet:MessageLimit>
</tet:PullMessages>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

15.10.6 拉消息应答

下面的拉取信息应答包含匹配这个订阅的两个通知。应答通知客户两个对象已经越过了“MyImportantFence1” and “MyImportantFence2 规则相关的线。

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
```

```
xmlns:wsa="http://www.w3.org/2005/08/addressing"
xmlns:wstop="http://docs.oasis-open.org/wsn/t-1"
xmlns:wsnt=http://docs.oasis-open.org/wsn/b-2
xmlns:tet="http://www.onvif.org/ver10/events/wsdl"
xmlns:tns1="http://www.onvif.org/ver10/topics"
xmlns:tt="http://www.onvif.org/ver10/schema">
<SOAP-ENV:Header>
<wsa:Action>
http://www.onvif.org/ver10/events/wsdl/PullPointSubscription/PullMessagesResponse
</wsa:Action>
</SOAP-ENV:Header>
<SOAP-ENV:Body>
<tet:PullMessagesResponse>
<tet:CurrentTime>
2008-10-10T12:24:58
</tet:CurrentTime>
<tet:TerminationTime>
2008-10-10T12:25:58
</tet:TerminationTime>
<wsnt:NotificationMessage>
<wsnt:Topic
Dialect="http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet">
tns1:RuleEngine/LineDetector/Crossed
</wsnt:Topic>
<wsnt:Message>
<tt:Message UtcTime="2008-10-10T12:24:57.321">
<tt:Source>
<tt:SimpleItem Name="VideoSourceConfigurationToken"
Value="1"/>
<tt:SimpleItem Name="VideoAnalyticsConfigurationToken"
Value="2"/>
<tt:SimpleItem Value="MyImportantFence1" Name="Rule"/>
</tt:Source>
<tt:Data>
<tt:SimpleItem Name="ObjectId" Value="15" />
</tt:Data>
</tt:Message>
</wsnt:Message>
</wsnt:NotificationMessage>
<wsnt:NotificationMessage>
<wsnt:Topic
Dialect="http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet">
tns1:RuleEngine/LineDetector/Crossed
</wsnt:Topic>
```

```
<wsnt:Message>
<tt:Message UtcTime="2008-10-10T12:24:57.789">
<tt:Source>
<tt:SimpleItem Name="VideoSourceConfigurationToken"
Value="1"/>
<tt:SimpleItem Name="VideoAnalyticsConfigurationToken"
Value="2"/>
<tt:SimpleItem Value="MyImportantFence2" Name="Rule"/>
</tt:Source>
<tt:Data>
<tt:SimpleItem Name="ObjectId" Value="19"/>
</tt:Data>
</tt:Message>
</wsnt:Message>
</wsnt:NotificationMessage>
</tet:PullMessagesResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

15.10.7 退订请求

客户端不得不明确的终止订阅通过使用一个可以立即释放资源的退订请求的设备。该请求是针对认购终端返回的，在创建 PullPoint 认购反应时候

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
xmlns:wsa="http://www.w3.org/2005/08/addressing"
xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2" >
<SOAP-ENV:Header>
<wsa:Action>
http://docs.oasis-open.org/wsn/bw-2/SubscriptionManager/UnsubscribeReques
t
</wsa:Action>
<wsa:To>http://160.10.64.10/Subscription?Idx=0</wsa:To>
</SOAP-ENV:Header>
<SOAP-ENV:Body>
<wsnt:Unsubscribe/>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

15.10.8 退订应答

一旦设备回复一个退订应答时，认购终端不再可用。

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
```

```
xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
xmlns:wsa="http://www.w3.org/2005/08/addressing"
xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2" >
<SOAP-ENV:Header>
<wsa:Action>
http://docs.oasis-open.org/wsn/bw-2/SubscriptionManager/UnsubscribeResponse
</wsa:Action>
</SOAP-ENV:Header>
<SOAP-ENV:Body>
<wsnt:UnsubscribeResponse/>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

15.11 服务错误码

除了定义在[WSBaseNotification].中的以外，事件服务没有定义任何具体的服务故障。。

16 PTZ 控制

PTZ 控制提供云台全方位（上下、左右）移动及镜头变倍、变焦控制。一个带有云台全方位移动及镜头变倍变焦能力的设备应该支持 PTZ 控制。类似的，一个具备变焦控制或全方位移动的设备也应该支持 PTZ 控制。[ONVIF PTZ WSDL]中详细定义 PTZ 控制。对应的命令操作说明请参考命令描述。

PTZ 控制覆盖了广泛的摄像设备。PTZ 设备可看作一个圆顶模型，可以改变摄像机的视角而不依赖于变焦控制

PTZ 的运动控制运用了坐标系模型，一个设备列出了一套他支持的坐标系统。规范提供了一套通用的可应用到任何 PTZ 设备的坐标空间，使进一步定义坐标系统成为了可能，使之更适用于圆顶设备。PTZ 控制应用于以下设备：

圆顶或云台摄像机

通过外部串口连接，带圆顶或 PTZ 摄像机的网络视频编码器

带数字 PTZ 的固定百万像素摄像机

带变焦的固定摄像机

PTZ 控制结构由 3 个主块组成：

PTZ Node – 一个底层的 PTZ 实体，用于管理 PTZ 设备和指定的特别功能

PTZ Configuration – 默认的坐标系统和对特定节点的默认的速度

PTZ Control Operation –移动，预设和辅助操作

PTZ 配置到配置文件中，PTZ 控制通过媒体配置文件（见 4.8.1），PTZ 的控制操作都是通过一个特定的配置文件来完成的。

PTZ 控制不提供创建或操作 PTZ 节点命令，对于每个可用的 PTZ 节点，应至少提供一个 PTZ 配置给这个配置节点，这个 PTZ 配置能够被添加到媒体配置文件中去，用于控制圆顶摄像机。每个媒体配置文件至多有一个 PTZ 配置。在媒体文件中，PTZ 配置和视频源配置被归为一类，因为视频源配置指的是通过 PTZ 配置控制的摄像机。

一旦存在的 PTZ 设备准备运行，一个具有 PTZ 功能的设备应该提供至少一个随时可用的

PTZ 配置文件，PTZ 配置文件应包含基本的设置和一个相应的视频源配置。

16.1 PTZ 模型

PTZ 模式把 PTZ 单元可能出现的运动分配给方向和变焦部分。通过控制 PTZ 单元，服务提供了绝对的，相对的和连续的运动控制。不同的坐标系和单元都用来支撑这些操作。
PTZ 服务提供了一个 AbsoluteMove 操作来移动 PTZ 设备到一个指定的位置，服务希望绝对的位置能作为一个参数定位到坐标系统中。镜头摇动的速度和变焦能被随意的指定，速度值是一个正的标量，它不能含有任何的方向信息。在没有当前位置的情况下，不能给摄像头的转动指定速度和变焦，否则它会产生一个不平滑和不直观的行为。

PTZ 服务中介绍了 RelativeMove 操作，用于控制圆顶摄像机移至相对位置，但是没有必要知道现在的位置。操作只需要一个位置参数，参数是根据引用的相对坐标系设置的。规范区分相对和绝对坐标系，是因为在某些情况下，一个定义好的相对坐标系没有绝对坐标系存在。一个可选的速度参数可以添加到 RelativeMove 操作中，它和绝对移动操作具有相同的意思。

最后，PTZ 设备可以通过 ContinuousMove 命令，使其在某一方向，某一速度下进行连续的移动。因此，一个速度矢量包含两个方面：方向和速度信息，后者通过矢量长度来表达通过适当的扩张坐标空间 URIs，云台坐标能够被唯一的指定,一个空间 URI 就表示一个根本的坐标系统。16.8 章节定义了一套坐标系的标准集。假如 PTZ 节点支持相对应类型的运动，那么 PTZ 节点就应该使用这些坐标系。在很多情况下，全方位位置移动就是说，在球形坐标系中摇动和倾斜一定的角度，一个操控固定百万像素摄像机的数字 PTZ, 通过静态的投影面上的一个像素位置，可以明确的知道摄像机的视角。因此，在这种情况下，为了获得 PTZ 设备物理的和虚拟的移动，需要使用不同的坐标系。这些和其他单独的坐标系被定义在一个独立的文档----ONVIF PTZ 坐标空间。为了使 NVCs 能充分利用 PTZ 节点的特性，一般，PTZ 节点可以定义它自己独有的一些特性到坐标系中。

通过 GetNode 或 GetNodes 操作，PTZ 节点可以重新获取寻 PTZ 节点描述，包含特定 PTZ 节点支持的所有坐标系。每一个坐标系属于以下分组中的其中之一：

AbsolutePanTiltPositionSpace 绝对移动位置空间
RelativePanTiltTranslationSpace 相对移动传输空间
ContinuousPanTiltVelocitySpace 连续移动速率空间
PanTiltSpeedSpace 移动速度空间
AbsoluteZoomPositionSpace 绝对变焦位置空间
RelativeZoomTranslationSpace 相对变焦传输空间
ContinuousZoomVelocitySpace 连续变焦速率空间

假如坐标系不支持以上某组中的坐标系，那么相应的移动操作就不适合 PTZ 节点。例如，假如列表不包含一个绝对移动位置空间，那么当指定一个绝对的位置移动时，绝对绝对移动操作就会失败。不同的空间就需要相对应的命令来描述。

16.2 PTZ 节点

一个具备 PTZ 功能的设备可以有多个 PTZ 节点。PTZ 节点可以表示机械的 PTZ 设备，上传 PTZ 设备或数字 PTZ 设备。在 PTZ 控制 API 中 PTZ 节点是最底层的实体，同时反映支持的 PTZ 功能。PTZ 节点的引用是通过它的名称或它的标识符，PTZ 服务不提供创建或操作 PTZ 节点的功能。

以下属性应该提供给所有的 PTZ 节点：

Token – 一个标识符，用于引用 PTZ 节点

Name –由安装者给的一个名字

SupportedPTZSpaces –一系列 PTZ 节点可用的坐标系，对于每一个坐标系，PTZ 应该指定它的允许范围

MaximumNumberOfPresets –假如一个预设被支持，那么所有的预设操作应该适用于这个节点

HomeSupported –一个布尔值指定一个起始位置的可用性，假如为真，对于 PTZ 节点起始位置操作应该可用的

AuxiliaryCommands –一系列支持的辅助命令。假如列表不为空，则辅助操作应该对 PTZ 节点可用。

16.2.1 获取所有节点（GetNodes）

一个可用的 PTZ 设备应该具备这个操作并返回这个设备上的所有可用的 PTZ 节点。

表 241：获取节点命令

获取节点 （GetNodes）	请求应答（Request-Response）
消息名字	描述
读取节点请求 (GetNodesRequest)	这是一个空消息
读取节点响应 （GetNodesResponse）	响应消息包含设备上一系列的 PTZ 节点 tt:PTZNode PTZNode[0][unbounded]
故障代码	描述
env:Receiver ter:ActionNotSupported ter:PTZNotSupported	设备不支持 PTZ

16.2.2 获取节点（GetNode）

假如存在一个具备 PTZ 功能的设备，那么它就能实现获取节点操作并返回被请求节点的特性，否则，设备应该返回一个适当的错误信息。

表 215：获取节点命令

获取节点 （GetNode） 请求应答（Request-Response）	
消息名字	描述
读取节点请求 (GetNodeRequest)	这个消息中包含了被请求的节点 tt:ReferenceToken NodeToken[1][1]
读取节点响应 （GetNodeResponse）	请求的 PTZNode 响应消息 tt:PTZNode PTZNode[0][unbounded]
故障代码	描述
env:Sender ter:InvalidArgVal ter:NoEntity	设备上没有这样的 PTZNode

env:Receiver ter:ActionNotSupported ter:PTZNotSupported	不支持 PTZ

16.3 PTZ 配置

PTZ 配置包含 PTZ 节点的归属。这个归属不能被一个 NVC 所改变。

以下元素是 PTZ 配置的一部分：

PTZNodeToken –配置 PTZ 节点的归属

DefaultAbsolutePanTiltPositionSpace –假如 PTZ 节点支持绝对全方位运动，那么在默认情况下，它应该指定一个绝对全方位位置空间

DefaultRelativePanTiltTranslationSpace –假如 PTZ 节点支持相对全方位运动，那么在默认情况下，它应该指定一个相对全方位转化空间

DefaultContinuousPanTiltVelocitySpace –假如 PTZ 节点支持连续的全方位运动，那么在默认情况下，它应该指定一个连续的全方位速率空间

DefaultPanTiltSpeedSpace –假如 PTZ 节点支持绝对的或相对的全方位运动，那么在默认情况下，它应该指定一个全方位速度空间

DefaultAbsoluteZoomPositionSpace –假如 PTZ 节点支持绝对的变焦，那么在默认情况下，它应该指定一个绝对的变焦位置空间

DefaultRelativeZoomTranslationSpace –假如 PTZ 节点支持相对的变焦，那么在默认情况下，它应该指定一个相对的变焦转换空间

DefaultContinuousZoomVelocitySpace –假如 PTZ 节点支持连续的变焦，那么在默认情况下，它应该指定一个连续的变焦速率空间

DefaultZoomSpeedSpace –假如 PTZ 节点支持绝对或相对的变焦，那么在默认情况下，它应该指定一个变焦速度空间

DefaultPTZSpeed –假如 PTZ 节点支持绝对或相对的 PTZ 移动，它应该指定一个默认的 PTZ 速度空间

DefaultPTZTimeout –假如 PTZ 节点支持连续的移动，它应该定义一个默认的超时在移动停止后

PanTiltLimits – 对于支持绝对全方位移动的 PTZ 设备，应该定义一个全方位移动限制机制。假如有这个机制，那么在配置中表示支持全方位移动限制机制。假如限制使能，那么表示全方位移动应保持在规定的范围内，全方位不使能通过设置–INF 或 +INF 实现。

ZoomLimits –对于支持绝对变焦的 PTZ 设备，应该定义一个变焦限制机制。假如有这个机制，那么在配置中表示支持变焦限制机制。假如限制使能，那么表示变焦应保持在规定的范围内，变焦不使能通过设置–INF 和+INF 实现。

在 NVC 没有指定一个坐标系的情况下，默认的位置/平移/速率空间允许发送移动请求，在没有明确的速度设定下，默认的速度为控制移动请求（绝对的，相对的，预设的）的速度。云台全方位移动限制的移动范围是通过一个二维空间范围设定设定，它能映射一个指定的绝对移动位置空间，对于支持云台全方位移动的 PTZ 节点，至少需要一个全方位移动位置空间，这个限制适用于所有的被支持的绝对，相对和连续的全方位移动。对于设定了限制的摄像头应该在坐标系内检查限制，意思就是说，即使在不同的坐标系中指定移动操作，但在最后，请求的移动都应该转换到已做过限制监测的坐标系中。当指定的一个相对或连续的移动操作需要超出指定的限制范围，那么 PTZ 单元就必须扩大指定的限制范围，变焦限制也必须相应的调整。

16.3.1 读取所有配置命令（GetConfigurations）

通过 GetConfigurations 命令操作，一个具有 PTZ 功能的设备应该能返回所有有效的 PTZ 配置。

表 216：GetConfigurations 命令

读取配置（GetConfigurations）	请求应答（Request-Response）
消息名字	描述
读取所有配置请求 (GetConfigurations)	这是一个空消息
读取所有配置响应 （GetConfigurationsResponse）	响应消息包含所有存在的 PTZConfiguration tt:PTZConfiguration PTZConfiguration [0][unbounded]
故障代码	描述
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	设备不支持 PTZ

16.3.2 读取配置命令（GetConfiguration）

假如存在一个具备 PTZ 功能的设备，那么通过 GetConfiguration 命令操作，它就能返回请求的 PTZ 配置。

表 217：读取配置命令

获取节点（GetConfiguration）	请求应答（Request-Response）
消息名字	描述
读取配置请求 (GetConfigurationRequestt)	这个消息中包含了被请求的 PTZ 配置 tt:ReferenceToken ConfigurationToken[1][1]
读取配置响应 （GetConfigurationResponse）	响应消息包含被请求的 PTZ 配置 tt:PTZConfiguration PTZConfiguration [1][1]
故障代码	描述
env:Sender ter: InvalidArgVal ter:NoConfig	被请求的配置不存在
env:Receiver ter:ActionNotSupported ter:PTZNotSupported	设备不支持 PTZ

16.3.3 读取配置选项（GetConfigurationOptions）

一个具备 PTZ 功能的设备应该支持 GetConfigurationOptions 操作。它返回一系列支持的坐标系列表，包括它们的范围限制。因此，这个选项可能不同，依赖于是否 PTZ 配置被分配给一个包含视频源配置的配置文件。在这种情况下，选项可以包含被视频源配置描述的影像坐标系。每一个列出的坐标系都属于 16.1 章列出的分组之一。假如 PTZ 节点支持连续的移动，它应该返回一个在 PTZ 节点能接受的超时范围内。

表 218：读取配置选项命令

获取配置选项（GetConfigurationOptions）	请求应答（Request-Response）
消息名字	描述
读取配置选项请求 (GetConfigurationOptions-Request)	这个消息中包含了 PTZ 配置标号 ConfigurationToken 指定打算使用的配置 Tt:ReferenceToken ConfigurationToken [1][1]
读取配置选项响应 （GetConfigurationOptions-Response）	响应消息包含配置选项 tt:PTZConfigurationOptionsPTZConfigurationOptions[1][1]
故障代码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	被请求的配置不存在
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	设备不支持 PTZ

16.3.4 设置配置（SetConfiguration）

一个具备 PTZ 功能的设备应该能实现 SetConfiguration 操作。强制保持标志(ForcePersistenceFlag)表示在重启设备后改变是否保持。

表 219：设置配置命令

设置配置（SetConfiguration）	请求应答（Request-Response）
消息名字	描述
设置配置请求 (SetConfigurationRequest)	PTZConfiguration 元素包含改进的 PTZ 配置，配置应该存在于设备中。 ForcePersistence 命令决定改变是否存储和在重启后改变是否保持，假如为真，那么在重启后改变保持，假如为假，那么在重启后改变可能回归到先前的值 tt:PTZConfiguration PTZConfiguration[1][1] xs:boolean ForcePersistence[1][1]
设置配置响应 （SetConfigurationResponse）	这是一个空消息
故障代码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	被请求的配置不存在
env:Sender ter:InvalidArgVal ter:ConfigModify	配置参数不适合
env:Receiver ter:Action ter:ConfigurationConflict	新的配置与其他配置冲突
env: Receiver	设备不支持 PTZ

ter:ActionNotSupported ter:PTZNotSupported	

16.4 移动操作

这个部分描述了三个操作，绝对地，相对地或连续地移动 PTZ 装置。所有的操作需要一个 ProfileToken，ProfileToken 用于定位含有一个 PTZ 配置的媒体配置文件。
所有的移动命令是非阻塞的，意思是说它们不用等到请求移动操作完成、最后一个移动操作可以被覆盖通过送一个新的移动请求。

16.4.1 绝对的移动（AbsoluteMove）

假如 PTZ 节点支持绝对的全方位和变焦移动，那么它应该支持 AbsoluteMove 操作。这个命令的位置参数指定了 PTZ 装置移动的绝对位置。它分裂成一个全方位移动操作元素和一个变焦操作元素。假如全方位位置被省略，那么通过这个命令，现在的位置应该不受影响。这个同样的适用于变焦操作。

涉及的位置空间应该是 PTZ 节点支持的绝对位置空间。假如空间信息被省略，那么应该使用 PTZ 配置默认的空间，PTZ 配置属于指定媒体配置文件的一部分。通过只给支持的情况提供绝对移动位置空间，一个设备可以支持绝对全方位移动、绝对变焦移动或没有绝对移动。在移动到请求位置过程中， PTZ 配置的默认速度参数可以被重写，假如空间引用了速度参数，那么 PTZ 节点就应该支持速度空间。
假如请求绝对位置没有被获得，那么操作就会失败。

表 220：绝对移动命令

绝对移动（AbsoluteMove）	请求应答（Request-Response）
消息名字	描述
绝对移动 (AbsoluteMove)	这个消息包含媒体文件的引用，一个用于指定目标位置的 Position 矢量和一个可选的 Speed tt:ReferenceToken ProfileToken [1][1] tt:PTZVector Position [1][1] tt:PTZSpeed Speed [0][1]
绝对移动响应 （AbsoluteMoveResponse）	这是一个空消息
故障代码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	请求的文件标识符 ProfileToken 不存在
env:Sender ter:InvalidArgVal ter:NoPTZProfile	请求的文件标识符不涉及 PTZ 配置
env:Sender ter:InvalidArgVal ter:SpaceNotSupported	空间引用了一个 PTZ 节点不支持的参数
env:Sender ter:InvalidArgVal	请求的位置越界

ter:InvalidPosition	
env:Sender ter:InvalidArgVal ter:InvalidSpeed	请求的速度越界
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	不支持 PTZ

16.4.2 相对移动（RelativeMove）

假如 PTZ 节点支持相对的全方位或相对的变焦移动，那么它应该支持 RelativeMove 操作，这个操作的意思就是说从当前位置移至 PTZ 设备想要到达的位置。操作分为方位元素和变焦元素。假如方位元素被省略，那么命令应该对现在的位置没有影响。同样适用于变焦元素。转换元素空间应该是被节点支持的转换空间。假如对于转换参数，空间信息被省略，那么媒体文件中的 PTZ 配置对应的默认空间会被使用。通过只给支持的情况提供绝对转换空间，一个设备可能支持相对全方位移动，相对变焦移动或没有相对移动。

在请求转换移动过程中， PTZ 配置的默认速度参数可以被重写，假如空间引用在速度参数之内，它们应该是被 PTZ 节点支持的速度空间。

通过送 0 值给全方位移动和变焦，命令可以使 PTZ 装置停止在当前位置。停止应该有完全相同的影响，而不依赖于涉及的相对空间。

假如被请求的转换通向一个不能到达的绝对的位置，PTZ 节点应移动到可达到的正确位置的边缘。

表 221：相对移动命令

相对移动（RelativeMove）	请求应答（Request-Response）
消息名字	描述
相对移动请求 (RelativeMoveRequest)	这个消息引用了一个媒体文件，一个相对于当前位置的位置 Translation 参数和一个可选的 Speed 参数 tt:ReferenceToken ProfileToken [1][1] tt:PTZVector Translation [1][1] tt:PTZSpeed Speed [0][1]
相对移动响应 （RelativeMoveResponse）	这是一个空消息
故障代码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	请求的文件标识符 ProfileToken 不存在
env:Sender ter:InvalidArgVal ter:NoPTZProfile	请求的文件标识符不涉及 PTZ 配置
env:Sender ter:InvalidArgVal ter:SpaceNotSupported	空间涉及了一个 PTZ 节点不支持的参数
env:Sender	请求的转换位置越界

ter:InvalidArgVal er:InvalidTranslation	
env:Sender ter:InvalidArgVal ter:InvalidSpeed	请求的速度越界
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	不支持 PTZ

16.4.3 连续移动（ContinuousMove）

一个具备 PTZ 功能的设备应该支持连续移动。这个命令的速率参数指定一个已确定的速度值给摄像头的移动和变焦。方位控制和变焦控制都是可选择的，假如方位控制元素被忽略，那么命令将不会对方位移动造成影响。这个同样适用于变焦移动。涉及到速率元素的空间应该是被 PTZ 节点支持的速率空间，假如空间信息忽略了速率参数，那么就会使用相对应的配置文件中 PTZ 配置里的默认参数，通过对支持的事件提供速率空间，一个设备可以支持连续全方位移动，连续变焦移动。

PTZ 配置中的默认超时参数可以被重写。超时参数指定 PTZ 节点连续移动的时长。通过送 0 值给全方位移动和变焦，命令可以使 PTZ 装置停止在当前位置。停止应该有完全相同的影响，而不依赖于涉及的速率空间。这个命令有同样的影响在连续移动中，就像在 16.4.4 章节中指定的停止命令。

假如被请求的速率通向一个不能到达的绝对的位置，PTZ 节点应移动到可达到的位置边缘。连续移动的一个典型应用是通过控制杆控制 PTZ。

表 222：连续移动命令

连续移动（ContinuousMove）	请求应答（Request-Response）
消息名字	描述
连续移动请求 (ContinuousMoveRequest)	这个消息引用了一个媒体文件，一个速率 Velocity 参数矢量指定方位移动和变焦的速率和一个可选的超时 Timeout 参数 tt:ReferenceToken ProfileToken [1][1] tt:PTZSpeed Velocity [1][1] xs:duration Timeout [0][1]
连续移动响应 （ContinuousMoveResponse）	这是一个空消息
故障代码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	请求的文件标识符 ProfileToken 不存在
env:Sender ter:InvalidArgVal ter:NoPTZProfile	请求的文件标识符不涉及 PTZ 配置
env:Sender ter:InvalidArgVal	空间涉及了一个 PTZ 节点不支持的参数

ter:SpaceNotSupported	
env:Sender ter:InvalidArgVal ter:TimeoutNotSupported	请求的超时越界
env:Sender ter:InvalidArgVal ter:InvalidVelocity	请求的速率越界
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	不支持 PTZ

16.4.4 停止（Stop）

一个具备 PTZ 功能的设备应该支持停止操作，假如停止命令被执行，那么命令会停止所有进行中的方向和变焦运动。通过设定相对应的参数，停止命令可以实现只停止某个特定的动作。

表 223：停止命令

连续移动（Stop）	请求应答（Request-Response）
消息名字	描述
停止请求 (StopRequest)	这个消息涉及到一个媒体文件和停止参数 tt:ReferenceToken ProfileToken[1][1] xs:boolean PanTilt [0][1] xs:boolean Zoom0][1]
停止响应 （StopResponse）	这是一个空消息
故障代码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	请求的文件标识符 ProfileToken 不存在
env:Sender ter:InvalidArgVal ter:NoPTZProfile	请求的文件标识符不涉及 PTZ 配置
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	不支持 PTZ

16.4.5 读取状态（GetStatus）

一个具有 PTZ 功能的设备应该能通过读取状态（GetStatus）命令报告 PTZ 状态。PTZ 状态包含以下信息：

Position (optional) –指定 PTZ 装置的绝对参数，相应的 PTZ 配置的默认绝对空间应该在位置元素之内被应用。

MoveStatus (optional) –表示是否云台设备是正在移动，闲置或未知状态

Error (optional) –发生一个 PTZ 错误
UTC Time –当这个状态产生时，指定实时时间

表 224：读取状态

读取状态（GetStatus）	请求应答（Request-Response）
消息名字	描述
读取状态请求 (GetStatusRequest)	这个消息引用了一个媒体文件，请求 PTZ 状态 tt:ReferenceToken ProfileToken[1][1]
读取状态响应 （GetStatusResponse）	这个消息包含一个被请求配置文件的状态 PTZStatus tt:PTZStatus PTZStatus[1][1]
故障代码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	请求的文件不存在
env:Sender ter:InvalidArgVal ter:NoPTZProfile	请求的文件标记不涉及 PTZ 配置
env:Receiver ter:Action ter:NoStatus	在媒体文件中没有适合的状态
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	不支持 PTZ

16.5 起始位置操作

这个部分描述如何管理 PTZ 节点的预设，对于支持预设的 PTZ 节点都应该能够执行这个操作，所有的操作需要一个文件标识符，文件标识符在媒体文件中 PTZ 配置里。

16.5.1 设置预设值（SetPreset）

设置预设命令是保存当前设备的位置参数，以便通过 GotoPreset 命令使设备能够移动到保存的预设位置。

为了能创建一个新的预设， SetPresetRequest 没有包含预设标识符。假如创建成功，响应就会反馈一个唯一的标识符。通过相应的预设标识符，预设可以被重新设定。在重写或创建两种情况下，一个可选的预设名(PresetName)可以被指定,假如在设置预设操作时 PTZ 设备正在移动，那么操作就会失败。设备在 PTZ 预设内可以保存额外的状态属性如成像参数，然后可以通过 GotoPreset 操作可取回。

表 225：设置预设命令

设置预设（SetPreset） （Request-Response）	请求应答
消息名字	描述
设置预设请求 (SetPresetRequest)	这个消息引用了一个媒体文件和被请求的预设名称 PresetName 或标识符 PresetToken

	tt:ReferenceToken ProfileToken[1][1] tt:ReferenceToken PresetToken[0][1] xs:string PresetName[0][1]
设置预设响应 （SetPresetResponse）	消息返回被设置的预设 tt:ReferenceToken PresetToken[1][1]
故障代码	描述
env:Sender ter:InvalidArgVal ter:PresetExist	被要求的名称已存在另一个预设中
env:Sender ter:InvalidArgVal ter:InvalidPresetName	预设名称过长或包含一个无效的字符
env:Receiver ter:Action ter:Moving	预设不能被设定当 PTZ 装置在移动过程中
env:Receiver ter:Action ter:TooManyPresets	达到预设的最大值
env:Sender ter:InvalidArgVal ter:NoProfile	请求的文件标识符 ProfileToken 不存在
env:Sender ter:InvalidArgVal ter:NoToken	请求的文件标识符不存在
env:Sender ter:InvalidArgVal ter:NoPTZProfile	请求的文件标识符不涉及 PTZ 配置
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	设备不支持 PTZ

16.5.2 读取所有预设值（GetPresets）

GetPresets 命令返回保存的预设值，由以下元素组成：

Token –标识符，对应预设的唯一识别符

Name –名称，一个可选的助记符

PTZ Position –PTZ 位置，一个可选的绝对位置，假如 PTZ 节点支持绝对的方位位置空间，那么全方位位置应该被指定。假如 PTZ 节点支持绝对的变焦位置空间，那么变焦位置应该被指定。

表 226：读取预设命令

读取预设（GetPresets）	请求应答（Request-Response）
消息名字	描述
读取预设请求	这个消息引用了一个媒体文件，用于告知请求读取的位

(GetPresetsRequest)	置 tt:ReferenceToken ProfileToken[1][1]
读取预设响应 （GetPresetsResponse）	消息包含一系列所请求媒体文件的预设值 Preset tt:PTZPreset Preset[0][unbounded]
故障代码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	请求的文件标识符 ProfileToken 不存在
env:Sender ter:InvalidArgVal ter:NoPTZProfile	请求的文件标识符不涉及 PTZ 配置
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	不支持 PTZ

16.5.3 返回预设

GotoPreset 命令召回先前的预设，假如速度参数被忽略，那么相应的 PTZ 配置中的默认速度就会被使用。对于 PTZ 节点，速度参数仅能在速度空间可用的情况下被指定。GotoPreset 命令是一个非阻塞操作，它能够被其他移动命令所中断。

表 227：转到预设命令

返回预设（GotoPreset） （Request-Response）	请求应答
消息名字	描述
返回预设请求 (GotoPresetRequest)	这个消息涉及到一个媒体文件，通过标识符鉴定移动至预设要发生的位置 tt:ReferenceToken ProfileToken[1][1] tt:ReferenceToken PresetToken[1][1] tt:PTZSpeed Speed[0][1]
返回预设响应 （GotoPresetResponse）	空消息
故障代码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	请求的文件标识符 ProfileToken 不存在
env:Sender ter:InvalidArgVal ter:NoToken	请求的文件标识符不存在
env:Sender ter:InvalidArgVal ter:SpaceNotSupported	PTZ 节点不支持的一个空间
env:Sender ter:InvalidArgVal	请求的文件标识符不涉及 PTZ 配置

ter:NoPTZProfile	
env:Sender ter:InvalidArgs ter:InvalidSpeed	要求的速度超出范围
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	设备不支持 PTZ

16.5.4 移除预设（RemovePreset）

RemovePreset 命令是删除先前设定的预设。

表 228：删除预设命令

移除预设（RemovePreset）	请求应答（Request-Response）
消息名字	描述
移除预设请求 (RemovePresetRequest)	这个消息涉及到一个媒体文件，通过标识符告知删除哪个被定义的预设 tt:ReferenceToken ProfileToken[1][1] tt:ReferenceToken PresetToken[1][1]
移除预设响应 （RemovePresetResponse）	空消息
故障代码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	请求的文件标识符 ProfileToken 不存在
env:Sender ter:InvalidArgVal ter:NoToken	请求的文件标识符不存在
env:Sender ter:InvalidArgVal ter:NoPTZProfile	请求的文件节点不涉及 PTZ 配置
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	设备不支持 PTZ

16.6 归位点操作

这章描述的操作用于管理 PTZ 节点的归位点。这些操作应用于支持归位点操作的 PTZ 节点。所有的操作需要一个文件标识符来定位一个媒体文件包括一个 PTZ 配置。
“原先”位置可以通过设置归位(SetHome)操作设定或是一个固定位置的 PTZ 装置。

16.6.1 转到归位点（GotoHomePosition）

这个操作移动摄像机到他的原先位置，假如速度参数是忽略的，那么就用相对应 PTZ

配置中的默认速度，对于 PTZ 节点，速度参数只能在速度空间可用的情况下被指定。命令是非阻塞的，他能被其它的移动命令所中断。

表 229：转到归位点命令

转至归位点（GotoHomePosition）	请求应答（Request-Response）
消息名字	描述
转至归位点请求 (GotoHomePositionRequest)	这个消息涉及到一个媒体文件，告知操作发生位置 tt:ReferenceToken ProfileToken[1][1] tt:PTZSpeed Speed[0][1]
转至归位点响应 （GotoHomePositionResponse）	空消息
故障代码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	请求的文件标识符 ProfileToken 不存在
env:Receiver ter:Action ter:NoHomePosition	请求的文件没有定义中心位置
env:Sender ter:InvalidArgVal ter:NoPTZProfile	请求的文件节点不涉及 PTZ 配置
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	设备不支持 PTZ

16.6.2 设置归位点（SetHomePosition）

SetHomePosition 操作保存当前的位置参数作为归位点，以便归位点操作能够请求设备移至原先位置。

假如“原先”位置是固定的，并且不能被重写，那么设置归位点命令应该返回一个失败标志；假如设置归位点成功，那么用 GotoHomePosition 命令它应该能移至原先位置。

表 230：设置归位点命令

设置归位点（SetHomePosition）	请求应答（Request-Response）
消息名字	描述
设置归位点请求 (SetHomePositionRequest)	这个消息涉及到一个媒体文件，告知归位点设置位置 tt:ReferenceToken ProfileToken[1][1]
设置归位点响应 （SetHomePositionResponse）	空消息
故障代码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	请求的文件标识符 ProfileToken 不存在
env:Sender ter:InvalidArgVal	请求的文件标识符不涉及 PTZ 配置

ter:NoPTZProfile	
env:Receiver ter:Action ter:CannotOverwriteHome	归位点是固定的，不能被修改
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	设备不支持 PTZ

16.7 辅助操作

这章描述管理 PTZ 节点的辅助命令，如一个红外灯，一个加热器或一个刮水器。PTZ 节点应该支持这些操作，在节点属性中表明这些辅助命令。在 PTZ 配置文件中的媒体文件中，所有的操作，需要一个文件标识符。

16.7.1 发送辅助命令（SendAuxiliaryCommand）

这个操作用于调用设备上的一个辅助命令，支持的命令可以通过 PTZ 节点属性查找到。辅助命令应该和 PTZ 节点支持的命令列表相匹配；而不支持其它的辅助命令。假如 PTZ 节点列有 irlampon 命令，那么辅助命令参数应该是 irlampon，，当 PTZ 节点支持辅助命令的时候，SendAuxiliaryCommand 应该被响应。

表 231：送辅助命令

送辅助命令（SendAuxiliaryCommand）	请求应答（Request-Response）
消息名字	描述
送辅助命令请求 (SendAuxiliaryCommandRequest)	这个消息涉及到一个媒体文件，告知辅助请求位置和辅助请求数据 tt:ReferenceToken ProfileToken[1][1] tt:AuxiliaryData AuxiliaryData[1][1]
送辅助命令响应 （SendAuxiliaryCommandResponse）	消息包含辅助响应 tt:AuxiliaryData AuxiliaryResponse[1][1]
故障代码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	请求的文件标识符 ProfileToken 不存在
env:Sender ter:InvalidArgVal ter:NoPTZProfile	请求的文件标识符不涉及 PTZ 配置
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	不支持 PTZ

16.8 预定 PTZ 空间

空间是用来指定绝对的，相对的和连续的移动。然而绝对的移动需要一个绝对的位置，

相对的移动通过一个位置转换来指定，连续移动需要一个速率的规范（相对运动超时），对这三种情况，应用不同的坐标系用于描述想要的移动。泛化空间不会指定基本的 PTZ 模型，以便它能够应用于任何的 PTZ 硬件。附加的空间被定义在 ONVIF PTZ 坐标空间文档中。

16.8.1 绝对的位置空间

16.8.1.1 泛化的全方位移动空间

所有支持绝对方位移动的 PTZ 节点，都应该提供泛化的全方位位置空间，因为他不涉及到一个指定的物理范围。反之，当一个全范围的 PTZ 节点，在接下来的空间描述中，范围规格化为-1 到 1 时，那么范围应该被定义。

```
<tt:AbsolutePanTiltPositionSpace>
<tt:SpaceURI>
http://www.onvif.org/ver10/tptz/PanTiltSpaces/PositionGenericSpace
</tt:SpaceURI>
<tt:XRange>
<tt:Min>-1.0</tt:Min>
<tt:Max>1.0</tt:Max>
</tt:XRange>
<tt:YRange>
<tt:Min>-1.0</tt:Min>
<tt:Max>1.0</tt:Max>
</tt:YRange>
</tt:AbsolutePanTiltPositionSpace>
```

16.8.1.2 泛化的变焦位置空间

所有支持绝对变焦的 PTZ 节点都应该提供泛化的变焦位置空间，因为他不涉及指定的物理范围。反之，当全范围的变焦范围被规格化为 0（最短焦距）到 1（最长焦距）时，则应该定义范围。没有一个规定关于泛化的变焦范围是怎样管理变焦的放大倍率，视场或物理的变焦尺寸。这个结果在接下来的空间描述中：

```
<tt:AbsoluteZoomPositionSpace>
<tt:SpaceURI>
http://www.onvif.org/ver10/tptz/ZoomSpaces/PositionGenericSpace
</tt:SpaceURI>
<tt:XRange>
<tt:Min>0.0</tt:Min>
<tt:Max>1.0</tt:Max>
</tt:XRange>
</tt:AbsoluteZoomPositionSpace>
```

16.8.2 相对的转换空间

一个相对的方位转换空间是使 PTZ 装置在某个指定方向上移动一定的位置，而不用知道摄像机当前位置。

16.8.2.1 泛化的方位转换空间

所有支持相对方位移动的 PTZ 节点都应该提供泛化的方位转换空间，因为他不涉及指定的物理范围。反之，则应该定义范围，当全部正反转换范围被规格化为-1 到 1 时，也就是说正的转换将会表示顺时针转或在向右/上的方向移动，如同接下来的空间描述：

```
<tt:RelativePanTiltTranslationSpace>
<tt:SpaceURI>
http://www.onvif.org/ver10/tptz/PanTiltSpaces/TranslationGenericSpace
</tt:SpaceURI>
<tt:XRange>
<tt:Min>-1.0</tt:Min>
<tt:Max>1.0</tt:Max>
</tt:XRange>
<tt:YRange>
<tt:Min>-1.0</tt:Min>
<tt:Max>1.0</tt:Max>
</tt:YRange>
</tt:RelativePanTiltTranslationSpace>
```

16.8.2.2 泛化的变焦转换空间

所有支持相对变焦的 PTZ 节点都应该提供泛化的变焦转换空间，因为他不涉及指定的物理范围。反之，当变焦的全正方转换范围被规格化为-1 到 1 时，类似的绝对范围应该被定义，就是在焦距方向，一个位置转换管理一个移动。转换的符号表示方向（负号表示宽度，正号表示焦距方向），没有一个规定关于泛化的变焦范围是怎样管理变焦的放大倍率，视场或其他物理的变焦尺寸。这个结果在接下来的空间描述中：

```
<tt:RelativeZoomTranslationSpace>
<tt:SpaceURI>
http://www.onvif.org/ver10/tptz/ZoomSpaces/TranslationGenericSpace
</tt:SpaceURI>
<tt:XRange>
<tt:Min>-1.0</tt:Min>
<tt:Max>1.0</tt:Max>
</tt:XRange>
</tt:RelativeZoomTranslationSpace>
```

16.8.3 连续的速率空间

连续的速率空间用于在一个指定方向上连续移动 PTZ 装置。

16.8.3.1 泛化的方位速率空间

所有的 PTZ 节点都应该被提供泛化的方位速率空间，因为他不涉及指定的物理范围。反之，当 PTZ 装置的速度范围被规格化为-1 到 1 时，也就是一个正的速率将会管理顺时针或在右/上的方向移动时，则范围应该被定义。一个带符号的速度能够单独的来指定给接下来在空间描述中的方位组件：

```
<tt:ContinuousPanTiltVelocitySpace>
<tt:SpaceURI>
http://www.onvif.org/ver10/tptz/PanTiltSpaces/VelocityGenericSpace
</tt:SpaceURI>
<tt:XRange>
<tt:Min>-1.0</tt:Min>
<tt:Max>1.0</tt:Max>
</tt:XRange>
<tt:YRange>
<tt:Min>-1.0</tt:Min>
<tt:Max>1.0</tt:Max>
</tt:YRange>
</tt:ContinuousPanTiltVelocitySpace>
```

16.8.3.2 泛化的变焦速率空间

在不知道具体的物理模型的情况下，泛化的变焦速率空间指定一个变焦速率。范围应该规范化为-1 到 1，就是一个正的速率管理焦距方向。一个泛化的变焦速率空间类似于接下来描述：

```
<tt:ContinuousZoomVelocitySpace>
<tt:SpaceURI>
http://www.onvif.org/ver10/tptz/ZoomSpaces/VelocityGenericSpace
</tt:SpaceURI>
<tt:XRange>
<tt:Min>-1.0</tt:Min>
<tt:Max>1.0</tt:Max>
</tt:XRange>
</tt:ContinuousZoomVelocitySpace>
```

16.8.4 速度空间

当摄像机移动到一个绝对或相对位置时，速度空间用于指定一个移动和变焦速度。与速率空间相比，速度空间不包含任何的方向信息，一个全方位移动的速度就是一个正数表示的标量。

16.8.4.1 泛化的方位速度空间

任何 PTZ 节点支持可配置的方位速度都应该提供泛化的方位速度空间，因为它不涉及一个指定的物理范围。反之，当全范围的速度范围规范化为 0（停止）到 1（最大速度）时，范围应该被定义。这个结果在接下来的空间描述中：

```
<tt:PanTiltSpeedSpace>
<tt:SpaceURI>
http://www.onvif.org/ver10/tptz/PanTiltSpaces/GenericSpeedSpace
</tt:SpaceURI>
<tt:XRange>
<tt:Min>0.0</tt:Min>
```

<tt:Max>1.0</tt:Max>
</tt:XRange>
</tt:PanTiltSpeedSpace>

16.8.4.2 泛化的变焦速度空间

所有支持可配置的变焦速度的 PTZ 装置，都应该提供泛化的变焦速度空间，因为他不涉及一个指定的物理范围。反之，当全速率范围被规范化为 0（停止）到 1（最大速度）时，范围应该被定义。这个结果在接下来的空间描述中：

<tt:ZoomSpeedSpace>
<tt:SpaceURI>
http://www.onvif.org/ver10/tptz/ZoomSpaces/ZoomGenericSpeedSpace
</tt:SpaceURI>
<tt:XRange>
<tt:Min>0.0</tt:Min>
</tt:XRange>
</tt:ZoomSpeedSpace>

16.9 服务错误码

表 232 列举的是 PTZ 服务错误码，每个命令产生一个类错误，见表 6.
一个特定的错误都被规定为一个类错误的子代码，见 5.11.2.1 章节，父类子代码是上一级错误码的子代码，具体的错误子代码在错误码的底层。

表 232：PTZ 具体错误码

错误码	父类子错误码 子错误码	错误原因	描述
env: 接收器（Receiver）	ter: Action	预设不能被设定	当 PTZ 装置在运动过中，预设不能被设置
	ter: MovingPTZ		
env: 接收器（Receiver）	ter:Action	数目达到了预设限制	预设达到了最大值
	ter:TooManyPresets		
env: 接收器（Receiver）	ter:ActionNotSupported	不支持 PTZ	设备不支持 PTZ
	ter: PTZNotSupported		
env: 发射机（Sender）	ter: InvalidArgVal	标识符已经存在	请求的标识符或名称已经存在于其他预设中
	ter:PresetExist		
env: 接收器	ter:Action	没有合适的 PTZ	在请求的媒体文件

（Receiver）		状态	中没有适合的 PTZ 状态
	ter: NoStatus		
env: 接收器（Receiver）	ter: Action	用新的配置是发生冲突	新的设置结果与配置不符
	ter:ConfigurationConflict		
env: 接收器（Receiver）	ter:Action	归位点不能被重写	归位点是固定的或不能被重写
	ter:CannotOverwriteHome		
env:发射机（Sender）	ter:InvalidArgVal	没有这样的 PTZ 节点	设备中没有这样的 PTZ 节点
	ter:NoEntity		
env:发射机（Sender）	ter:InvalidArgVal	没有这样的配置	没有这样的配置存在
	ter:NoConfig		
env:发射机（Sender）	ter:InvalidArgVal	参数不能被设置	配置参数可能不能被设置
	ter:ConfigModify		
env:发射机（Sender）	ter:InvalidArgVal	终点超出范围	请求的目标超出了范围
	ter:InvalidPosition		
env:发射机（Sender）	ter:InvalidArgVal	转换超出范围	请求的转换超出了范围
	ter:InvalidTranslation		
env:发射机（Sender）	ter:InvalidArgVal	请求速度超出范围	请求的速度超出了范围
	ter:InvalidSpeed		
env:发射机（Sender）	ter:InvalidArgVal	速率超出范围	请求的速率超出了范围

	ter:InvalidVelocity		
env:发射机（Sender）	ter:InvalidArgVal	预设名称太长	预设名称过长或包含了无效的字符
	ter:InvalidPresetNme		
env:发射机（Sender）	ter:InvalidArgVal	文件丢失 PTZ 配置	请求的文件标识符不能找到配置文件
	ter:NoPTZProfile		
env:发射机（Sender）	ter:InvalidArgVal	文件标识符不存在	请求的文件标识符 ProfileToken 不存在
	ter:NoProfile		
env:发射机（Sender）	ter:InvalidArgVal	不支持超时	指定的超时参数不在超时范围内
	ter:TimeoutNotSupported		
env:发射机（Sender）	ter:InvalidArgVal	标识符不存在	请求的标识符不存在
	ter:NoToken		
env: 接收器（Receiver）	ter:Action	没有归位点	这个文件没有归位点被设定
	ter:NoHomePosition		
env:发射机（Sender）	ter:InvalidArgVal	没有这样的空间	PTZ 节点不支持这个空间
	ter:SpaceNotSupported		

17 视频分析

4.12 节给我们简单地介绍了 ONVIF 视频分析结构，这章主要介绍结构的以下几个方面：

分析模块接口
场景描述
规则接口
时间接口

事件接口是通过事件服务处理的，事件服务在 15.17.1 节中介绍的基于 XML 场景描述部分有详细讲到，事件接口可以是流数据，就像元数据那样通过 RTP(详见 12.1.2.1.1)传送到客户端。媒体服务管理所有的规则引擎和分析引擎配置（见 11 章）。分析服务允许更多高密度的配置，就是说拥有个人独特的规则和个人独特的分析模块（详见 17.2 和 17.3）

一个支持分析的设备应该具备场景描述和事件接口，和媒体服务进行分析配置一样。假如设备还支持规则引擎，那么还应该在这个标准中定义分析引擎，然后它就能够实现规则分析模块接口。

通过媒体服务，一个完整的视频分析配置能够附属在文件上，一个视频分析配置可以指定到一个视频源（见 11.9 节）。由分析或规则引擎或当一个客户端请求类似的场景描述流时产生的事件，被客户端直接或间接地赞成时，设备应该确保相应的分析引擎开始工作。

17.1 场景描述接口

17.1.1 概述

这个规范定义了 XML 架构，用于设备进行编码场景描述，场景描述的范围包含基本场景元素到终端用户，就如同一个给供应商指定的扩展架构，附录 A.2 显示了一些附加的用于处理供应商处理规则的场景原理。

视频分析引擎通过媒体控制部分的文件来配置，假如在一个 Profile 中视频分析是有效的，VideoSourceConfiguration 和 VideoAnalyticsConfiguration 应该在文件中涉及，视频分析引擎通过参考

VideoSourceConfiguration 来处理画面。

17.1.2 画面相关内容

视频分析引擎的输入是视频源的图像。场景的提取原理和图像提取的地点息息相关，一个提取的场景不同于一般的被视频分析引擎（信息如视频输入行，视频分辨率，图像裁剪，帧率等）处理过的视频源， 当前图像与输入流相关联和在画面内进行元素空间定位。
视频源和视频分析部件之间的连接是媒体控制的一部分，允许视频分析器运行在减少帧率的剪切视频源上。

根据场景要素的暂态和空间关系来进行视频源的挑选，详见 17.1.2.1 和 17.1.2.2。跟踪目标的外官和行为，详见 17.1.3.1。目标的分离和合并，详见 17.1.3.2。

一个 PTZ 设备能够把方位和变焦信息放到一帧画面数据的开始，允许客户端评估 3D 坐标中的场景元素。接下来，图像坐标系可以用于一个可选的转换节点，转换节点用于下一个

分段的描述。最终，多重对象描述能够被置入，而且他们的关系能够被指定在一个目标树节点内。为了方便定义如下：

```
<xs:complexType name="Frame">
<xs:sequence>
<xs:element name="PTZStatus" type="tt:PTZStatus"
minOccurs="0"/>
<xs:element name="Transformation" type="tt:Transformation"
minOccurs="0"/>
<xs:element name="Object" type="tt:Object" minOccurs="0"
maxOccurs="unbounded"/>
<xs:element name="ObjectTree" type="tt:ObjectTree" minOccurs="0"/>
...
</xs:sequence>
<xs:attribute name="UtcTime" type="xs:dateTime" use="required"/>
...
</xs:complexType>
<xs:element name="Frame" type="tt:Frame">
```

请注意这里的模式仅供参考，【ONVIF 架构】包含了标准的框架定义。

17.1.2.1 章节描述了怎样进行图像处理，通过网络视频分析流中的视频分析算法。

17.1.2.1 时间关系

因为多重的场景元素能够从相同的图像中提取出来，场景元素都是基于一个帧节点的，能从视频输入链接到一个指定的影像。帧节点包含一个强制的时钟特性。这个时钟应该使能一个客户端来管理图像节点精确到一个视频图像。例如，相对应的 RTP 时标编码视频帧应该和 UTZ 时标转换的结果一致。视频和元数据流的同步性，在 12.1.2.2.1 章节实时查看部分有进一步描述。

例如：

```
<tt:Frame UtcTime="2008-10-10T12:24:57.321">
...
</tt:Frame>
...
<tt:Frame UtcTime="2008-10-10T12:24:57.521">
...
</tt:Frame>
```

17.1.2.2 空间关系

很多场景元素是提取出来的画面的一部分，例如，但随着时间追踪目标时，应该在每幅画面中指定跟踪目标的位置，这些位置应该和坐标系关联起来。默认的坐标系如图 23 中说明。管理选择的矩形相应 VideoSourceConfiguration 文件。

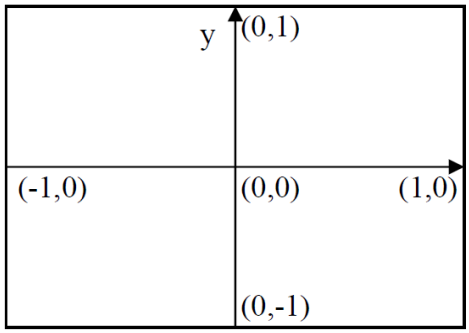


图 23：默认的画面坐标系

对于 XML 树的特定节点，规范允许修改其坐标系。因此，每个图像节点都是从默认的坐标系开始的。每个子节点继承最近的父类坐标系。一个转换节点用于修改最近的父类坐标系，相应的规范也与父类节点最近的坐标系相关联。

规范定义了转换节点用于缩放和转化，场景描述包含了这些转换节点存放的占位符。

```
<xs:complexType name="Transformation">
<xs:sequence>
<xs:element name="Translate" type="Vector" minOccurs="0"/>
<xs:element name="Scale" type="Vector" minOccurs="0"/>
...
</xs:sequence>
</xs:complexType>
```

规范遵照一个数字的坐标系和转换描述。一个坐标系由转换矢量 $t=\begin{pmatrix}t_x\\t_y\end{pmatrix}$ 和缩放矢量 $s=\begin{pmatrix}s_x\\s_y\end{pmatrix}$ 组成，通过公式 $\begin{pmatrix}q_x\\q_y\end{pmatrix}=\begin{pmatrix}p_x\cdot s_x+t_x\\p_y\cdot s_y+t_y\end{pmatrix}$，坐标系的一个点 $p=\begin{pmatrix}p_x\\p_y\end{pmatrix}$ 转换到默认的坐标系相对应的点 $q=\begin{pmatrix}q_x\\q_y\end{pmatrix}$，类似的，通过公式 $\begin{pmatrix}w_x\\w_y\end{pmatrix}=\begin{pmatrix}v_x\cdot s_x\\v_y\cdot s_y\end{pmatrix}$，坐标系给的矢量 v 可以转换到默认坐标系相应的矢量 w。

转换节点有一个可选的缩放矢量 $u=\begin{pmatrix}u_x\\u_y\end{pmatrix}$ 和一个可选的平移矢量 $v=\begin{pmatrix}v_x\\v_y\end{pmatrix}$。假如缩放比例不被指定，那么它默认的是 $u=\begin{pmatrix}1\\1\end{pmatrix}$，类似的，平移矢量的默认值是 $u=\begin{pmatrix}0\\0\end{pmatrix}$，转换节点通过用以下方法修改最高的坐标系：$\begin{pmatrix}t'_x\\t'_y\end{pmatrix}=\begin{pmatrix}v_x\cdot s_x+t_x\\v_y\cdot s_y+t_y\end{pmatrix}$，$\begin{pmatrix}s'_x\\s'_y\end{pmatrix}=\begin{pmatrix}u_x\cdot s_x\\u_y\cdot s_y\end{pmatrix}$，用 $\begin{pmatrix}t'_x\\t'_y\end{pmatrix}$ 和 $\begin{pmatrix}s'_x\\s'_y\end{pmatrix}$ 代替最高坐标系。

例如，场景描述的坐标是用图形坐标系来给的，左下角是坐标（0,0），右上角是（320,240）。

图形节点类似于以下代码，缩放比例设置为图像宽和高的 1/2.

```
<tt:Frame UtcTime="2008-10-10T12:24:57.321">
<tt:Transformation>
<tt:Translate x="-1.0" y="-1.0"/>
<tt:Scale x="0.00625" y="0.00834"/>
</tt:Transformation>
...
</tt:Frame>
```

17.1.3 场景元素

这个部分集中讲解场景元素，场景元素由对象跟踪算法，定义对象处理和对象图形组成。在场景描述内没有发现目标的图像可以被跳过以保存带宽，只要在场景描述中最后的图像也是空。即使场景描述是空，但还是建议设备定期的发送场景描述。为了验证分析设备正常工作。假如对应的流要求同步性，那么设备应该发送场景描述。

当场景描述的接收器接收到一副空的图像，接收器应该假设后续影像都是空的直到一个非空的影像被接收到。当最后接收的图像非空，那么接收器应该假设为下一副描述图像将会被传输。

17.1.3.1 对象

对象通过其 ID 进行辨认，一个特定对象的相关特征都被收集在带有相应对象 ID 的对象节点中。与对象相关联的，像对象重命名，对象的分裂，对象的合并和对象的删除，都存储在一个单独的对象树节点中。一个对象 ID 是对象节点内第一个被默认创建的特征。

```
<xs:complexType name="ObjectId">
<xs:attribute name="ObjectId" type="xs:int"/>
</xs:complexType>
<xs:complexType name="Object">
<xs:complexContent>
<xs:extension base="ObjectId">
<xs:sequence>
<xs:element name="Appearance" type="Appearance" minOccurs="0"/>
<xs:element name="Behaviour" type="Behaviour" minOccurs="0"/>
...
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>
```

对象节点有两个占位符用于外观和行为信息，外观节点从一个可选的转换节点开始，转换节点可用于改变一个中心图像坐标系到一个中心对象坐标系，然后，可以指定对象的形状。假如在画面中发现了一个对象，那么形状信息应该存在于外观描述，视频分析算法可以增加对象节点给现在不可见的对象，否则它不能为这个对象推断信息，在这样的情况下，形状描述可以被忽略。

其他的对象特征像颜色和对象级别可以被加到外观节点中去。这个规范集中于形状描述

(见 17.1.3.3)。颜色和对象级别的定义可以在附录 B.1 中找到。

这个规范定义了两个标准行为给对象。当一个对象停止移动的时候，它能够被标记为离开了或是静止的。这些行为应该被列出，作为一个对象的行为节点的子节点。一个离开的或静止节点的出现不会自动地删除相应的对象 ID,从而当同样的对象出现是可以再次利用对象 ID.

一个标有离开行为的对象指定了对象离开的位置，这个标记不应该用作离开对象的行为，尽管带走对象的行为没有被检测到，但还是有可能检测到移动，在运转中，先前的对象可能被标记作为静止的，表示对象停止了移动。一但这个对象不改变了，它们就将不被列在场景描述中。当一个静止的对象再次出现在场景描述中，静止标志就会自动的移除。

例如：

```
...
        <tt:Frame UtcTime="2008-10-10T12:24:57.321">
        <tt:Transformation>
        <tt:Translate x="-1.0" y="-1.0"/>
        <tt:Scale x="0.003125" y="0.00416667"/>
        </tt:Transformation>
        <tt:Object ObjectId="12">

        <tt:Appearance>
        <tt:Shape>
        <tt:BoundingBox left="20.0" top="30.0" right="100.0" bottom="80.0"/>
        <tt:CenterOfGravity x="60.0" y="50.0"/>
        </tt:Shape>
        </tt:Appearance>
        </tt:Object>
        </tt:Frame>
        ...
        <tt:Frame UtcTime="2008-10-10T12:24:57.421">
        <tt:Transformation>
        <tt:Translate x="-1.0" y="-1.0"/>
        <tt:Scale x="0.003125" y="0.00416667"/>
        </tt:Transformation>
        <tt:Object ObjectId="12">
        <tt:Appearance>
        <tt:Shape>
        <tt:BoundingBox left="20.0" top="30.0" right="100.0" bottom="80.0"/>
        <tt:CenterOfGravity x="60.0" y="50.0"/>
        </tt:Shape>
        </tt:Appearance>
        <tt:Behaviour>
        <tt:Idle/>
        </tt:Behaviour>
        </tt:Object>
        </tt:Frame>
```

```
...
<tt:Frame UtcTime="2008-10-10T12:24:57.521">
<tt:Transformation>
<tt:Translate x="-1.0" y="-1.0"/>
<tt:Scale x="0.003125" y="0.00416667"/>
</tt:Transformation>
</tt:Frame>
...
<tt:Frame UtcTime="2008-10-10T12:24:57.621">
<tt:Transformation>
<tt:Translate x="-1.0" y="-1.0"/>
<tt:Scale x="0.003125" y="0.00416667"/>
</tt:Transformation>
<tt:Object ObjectId="12">
<tt:Appearance>
<tt:Shape>
<tt:BoundingBox left="25.0" top="30.0" right="105.0" bottom="80.0"/>
<tt:CenterOfGravity x="65.0" y="50.0"/>
</tt:Shape>
</tt:Appearance>
</tt:Object>
</tt:Frame>
...
<tt:Frame UtcTime="2008-10-10T12:24:57.721">
<tt:Transformation>
<tt:Translate x="-1.0" y="-1.0"/>
<tt:Scale x="0.003125" y="0.00416667"/>
</tt:Transformation>
<tt:Object ObjectId="19">
<tt:Appearance>
<tt:Shape>
<tt:BoundingBox left="20.0" top="30.0" right="100.0" bottom="80.0"/>
<tt:CenterOfGravity x="60.0" y="50.0"/>
</tt:Shape>
</tt:Appearance>
<tt:Behaviour>
<tt:Removed/>
</tt:Behaviour>

</tt:Object>
</tt:Frame>
```

17.1.3.2 对象树

当两个对象走得非常接近的时候，例如视频分析可能不再单独的跟踪它们，一个目标合

并应该被告知通过增加一个合并节点到图像节点的对象树节点。一个合并节点包含一个合并对象 ID,列出了其来自节点和一个到节点，合并对象用于将来图像跟踪的 ID，假如视频分析算法察觉到一个对象正在阻塞其他对象，而将来有可能要跟踪这个对象，那么阻塞的对象应该放进到节点中。

分离的对象通过一个分裂节点来表示，在这种情况下，来自节点包含一个单一的对象，表示对象在现在的图像中是分裂的，由这个分裂对象分离出来的对象都被列在到节点中。来自节点的对象 ID 可能再次出现在到节点中，假如这个对象使其它的对象闭塞，视频分析算法可能跟踪这个对象在闭塞期间。

为了成为一个分裂操作的一部分，一个对象不需要涉及到一个分离操作。例如，假如一个对象正在移动连同一个人，人离开对象到某处，当人离开对象留在后面，对象可以首先通过视频分析被分离， 在这种情况下，对象的第一个外观可以是包含一个分裂的操作。
在后来，一副没有分裂的画面中，当一个合并的对象作为一个对象节点重新出现，那么这个对象会被软件分开。然而，视频分析算法不能决定分裂对象来自哪里。

一个视频分析算法能跟综和记住有限个对象，为了表明指定的对象已经从存储器中被删除并从在没在出现过，场景描述可以包含一个删除节点在对象树节点中
在执行一个分裂操作时，假如视频分析算法不能决定对象的 ID，那么应该给它使用一个新的对象 ID。当算法收集到足够关于这个对象的身份证明，它能够通过重命名操作改变对象的 ID,当一个对象重新出现在画面中或在一段时间后真正的 ID 丢失，重命名操作也可以被用。

一个删除的对象 ID 不能被重复使用，直到对象 ID 已经出项环绕时，才可以把使用过的对象 ID 重新使用在场景描述中。
例如：

```
<tt:Frame UtcTime="2008-10-10T12:24:57.321">
                    <tt:Object ObjectId="12">
                    ...
                    </tt:Object>
                    <tt:Object ObjectId="17">
                    ...
                    </tt:Object>
                    </tt:Frame>
                    <tt:Frame UtcTime="2008-10-10T12:24:57.421">
<tt:Object ObjectId="12">
                    ...
                    </tt:Object>
                    <tt:ObjectTree>
                    <tt:Merge>
                    <tt:From ObjectId="12"/>
                    <tt:From ObjectId="17"/>
                    <tt:To ObjectId="12"/>
                    </tt:Merge>
                    </tt:ObjectTree>
                    </tt:Frame>
                    <tt:Frame UtcTime="2008-10-10T12:24:57.521">
<tt:Object ObjectId="12">
                    ...
```

```
                </tt:Object>
                </tt:Frame>
                <tt:Frame UtcTime="2008-10-10T12:24:57.621">
<tt:Object ObjectId="12">
                ...
                </tt:Object>
                <tt:Object ObjectId="17">
                ...
                </tt:Object>
                <tt:ObjectTree>
                <tt:Split>
                <tt:From ObjectId="12"/>
                <tt:To ObjectId="17"/>
                <tt:To ObjectId="12"/>
                </tt:Split>
                </tt:ObjectTree>
                </tt:Frame>
```

17.1.3.3 形状描述符

形状信息应该放在对象外观节点中可选形状节点的下面，假如形状节点存在，形状节点保存信息在指定画面中被考虑对象检测到的地方，一个形状节点至少包含两个节点，就是边界框和检测对象的重心。

粗糙的边界框更进一步精确了附加的子节点，每一个都代表了一个形状原型。假如多边形基本体都定义了，那么它们的集合就可以定义对象的形状。在规范中提供了一个普通的多边形描述符号。

描述对象形状的多边形可以简化为一系列的点。

两个连续的点在列表中被定义为一条线段，点的组织应该认真筛选例如一个封闭的对象区域能够在左手边找到所有线段。多叉线的定义是通过一系列不自我交叉的点。

例如：

```
<tt:Frame UtcTime="2008-10-10T12:24:57.321">
                <tt:Transformation>
                <tt:Translate x="-1.0" y="-1".0/>
                <tt:Scale x="0.003125" y="0.00416667"/>
                </tt:Transformation>
                <tt:Object ObjectId="12">
                <tt:Appearance>
                <tt:Shape>
                <tt:BoundingBox left="20.0" top="30.0" right="100.0" bottom="80.0"/>
                <tt:CenterOfGravity x="60.0" y="50.0"/>
                <tt:Polygon>
                <tt:Point x="20.0" y="30.0"/>
                <tt:Point x="100.0" y="30.0"/>
                <tt:Point x="100.0" y="80.0"/>
                <tt:Point x="20.0" y="80.0"/>
```

```
</tt:Polygon>
</tt:Shape>
</tt:Appearance>
</tt:Object>
</tt:Frame>
```

17.2 规则接口

视频分析配置由两部分组成（见 11.9 章节），第一个部分配置创建场景描述的视频分析引擎。第二个部分配置规则引擎。对于第二部分，在 17.2.1 章节介绍了一个 XML 结构，用于规则配置的通信，在 17.2.2 章节指定了一种语言来描述特定规则类型配置，在 17.2.3 中定义了两个标准规则，用于设备执行规则引擎。17.2.4 章节介绍了规则的管理操作，假如设备支持一个规则引擎，它就应该支持所有的规则接口。

17.2.1 规则陈述

规则的配置有两个必要的属性：指定名称和指定规则类型。不同的配置参数被列在规则元素的参数元素下，每个参数是一个简单条款或一个元素条款（与消息负载的比较见 15 章）。在参数列表中，每一个名称属性都应该是独一不二的。简单条款有一个包含参数值的附加值属性。元素条款的值是通过元素条款的子元素给的。建议尽可能使用简单条款参数。
接下来的例子显示了一个完整的包含两个规则的视频分析配置：

```
<tt:VideoAnalyticsConfiguration>
<tt:AnalyticsEngineConfiguration>
...
</tt:AnalyticsEngineConfiguration>
<tt:RuleEngineConfiguration>
<tt:Rule Name="MyLineDetector" Type="tt:LineDetector">
<tt:Parameters>
<tt:SimpleItem Name="Direction" Value="Any"/>
<tt:ElementItem Name="Segments">
<tt:Polyline>
<tt:Point x="10.0" y="50.0"/>
<tt:Point x="100.0" y="50.0"/>
</tt:Polyline>
</tt:ElementItem>
</tt:Parameters>
</tt:Rule>
<tt:Rule Name="MyFieldDetector" Type="tt:FieldDetector">
<tt:Parameters>
<tt:ElementItem Name="Field">
<tt:Polygon>
<tt:Point x="10.0" y="50.0"/>
<tt:Point x="100.0" y="50.0"/>
<tt:Point x="100.0" y="150.0"/>
```

```
</tt:Polygon>
</tt:ElementItem>
</tt:Parameters>
</tt:Rule>
</tt:RuleEngineConfiguration>
</tt:VideoAnalyticsConfiguration>
```

17.2.2 规则描述语言

规则描述包含类型信息(属于一个中心规则类型的所有参数)，输出描述就是通过这个规则产生的。规则引擎的输出是一个事件，可以用于事件引擎或是客户端允许的事件。
一个中心规则类型的参数都被列在参数描述元素的下面。所有的参数是一个简单或元素条款，它们能够通过简单条款描述或一个元素条款来描述。两种条款描述包含一个名称属性和类型属性，名称属性用于鉴定参数，类型属性用于指定一种 XML 模式类型。在一个简单条款描述中，类型属性应定义为一种简单类型模式，在元素条款描述中，类型属性应声明为一种 XML 模式的全局元素。

由规则类型产生的输出被描述在多重消息描述元素中。每一个消息描述包含一个消息负载的描述，消息负载通过消息描述语言描述，详见 15 章。此外，消息描述应该包含一个父类元素，为了接收指定的输出，父类元素必须是一个客户端同意的对象，对象应该被指定作为一个具体的对象符号。

17.2.3 章节描述了规则描述语言在两个标准规则里的使用，为了方便定义如下：

```
<xs:element name="RuleDescription" type="tt:ConfigDescription"/>
<xs:complexType name="ConfigDescription">
<xs:sequence>
<xs:element name="ParameterDescription"
type="tt:ItemListDescription"/>
<xs:element name="Messages" minOccurs="0" maxOccurs="unbounded">
<xs:complexType>
<xs:complexContent>
<xs:extension base="tt:MessageDescription">
<xs:sequence>
<xs:element name="ParentTopic" type="xs:string"/>
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
...
</xs:sequence>
<xs:attribute name="Name" type="xs:string" use="required"/>
</xs:complexType>
<xs:complexType name="ItemListDescription">
<xs:sequence>
```

```
<xs:element name="SimpleItemDescription" minOccurs="0"
maxOccurs="unbounded">
<xs:complexType>
<xs:attribute name="Name" type="xs:string" use="required"/>
<xs:attribute name="Type" type="xs:QName" use="required"/>
</xs:complexType>
</xs:element>
<xs:element name="ElementItemDescription" minOccurs="0"
maxOccurs="unbounded">
<xs:complexType>
<xs:attribute name="Name" type="xs:string" use="required"/>
<xs:attribute name="Type" type="xs:QName" use="required"/>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
```

17.2.3 规则标准

以下的规则标准适用于静态的摄像机。在一个 PTZ 设备中，基于图像的规则应该包含一个附加的元素条款，元素条款用于识别设备规则安装位置。相应的元素条款描述类似如下：

```
<tt:ElementItemDescription Name="PTZStatus" Type="tt:PTZStatusType">
```

17.2.3.1 线性检测器

线性检测器定义为一个非交叉的简单多叉线，假如一个对象在一个指定的方向横跨多叉线，规范引擎就送一个交叉事件，交叉事件包含线性检测器的名称和涉及横跨交叉线的对象。方向，就是一个能够在左，右和其他方向选择的，方向左和右涉及到沿着线从一个点到另一个点走的方向和被禁止的方向。

线性检测器类似以下用规则描述语言的代码，详细的参考前一章：

```
<tt:RuleDescription Name="tt:LineDetector">
<tt:Parameters>
<tt:SimpleItemDescription Name="Direction" Type="tt:Direction"/>
<tt:ElementItemDescription Name="Segments" Type="tt:Polyline"/>
</tt:Parameters>
<tt:MessageDescription>
<tt:Source>
<tt:SimpleItemDescription Name="VideoSourceConfigurationToken"
Type="tt:ReferenceToken"/>
<tt:SimpleItemDescription Name="VideoAnalyticsConfigurationToken"
Type="tt:ReferenceToken"/>
<tt:SimpleItemDescription Name="Rule" Type="xs:string"/>
</tt:Source>
<tt:Data>
<tt:SimpleItemDescription Name="ObjectId" Type="tt:ObjectId"/>
```

```
</tt:Data>
        <tt:ParentTopic>tns1:RuleEngine/LineDetector/Crossed</tt:ParentTopic>
        </tt:MessageDescription>
        </tt:RuleDescription>
```

以上代码定义了两个参数，段数和方向，并使一个事件附属到主题：规则引擎/线性检测器/交叉相。

17.2.3.2 域检测器

一个域检测器被定义为一个简单的非交叉多边形。域检测器决定了一个画面中的对象在多边形的外面还是里面。这个信息会当成一个属性存放起来。

域检测器类似于以下代码，用规则描述语言详见上一章。

```
        <tt:RuleDescription Name="tt:FieldDetector">
        <tt:Parameters>
        <tt:ElementItemDescription Name="Field" Type="tt:Polygon"/>
        </tt:Parameters>
        <tt:MessageDescription IsProperty="true">
        <tt:Source>
        <tt:SimpleItemDescription Name="VideoSourceConfigurationToken"
Type="tt:ReferenceToken"/>
        <tt:SimpleItemDescription Name="VideoAnalyticsConfigurationToken"
        Type="tt:ReferenceToken"/>
        <tt:SimpleItemDescription Name="Rule" Type="xs:string"/>
        </tt:Source>
        <tt:Key>
        <tt:SimpleItemDescription Name="ObjectId" Type="tt:ObjectIdType"/>
        </tt:Key>
        <tt:Data>
        <tt:SimpleItemDescription Name="IsInside" Type="xs:boolean"/>
        </tt:Data >
        <tt:ParentTopic>
        tns1:RuleEngine/FieldDetector/ObjectsInside
        </tt:ParentTopic>
        </tt:MessageDescription>
        </tt:RuleDescription>
```

从内部属性看，一个客户端能够引导检测器参数的进入和离开。通过增加一个消息内容过滤器来认定，一个客户端能模拟出进入和离开事件，就是通过设定内部条款为真让内部对象消息通过。

17.2.4 规则操作

假如设备支持规范定义的规则引擎，那么它就具备以下管理规则的能力，创建/删除/修改命令具有原子性的特点，也就是说所有的操作要么都能被处理，要么都不能被处理。

17.2.4.1 读取支持的操作（GetSupportedRules）

设备应该输出所有支持的规则当执行随后的操作，它通过在 17.2.2 章节描述的规则描述语言，返回一个规则描述列表。此外，它包含了一系列 URLs 用于提供概要文件的位置。这些概要文件描述了在规则描述中用到的类型和元素。假如规则描述涉及到 ONVIF 概要文件中的类型和元素，ONVIF 概要文件应该明确地列出来。

表 233：读取支持的规则命令

读取支持的规则（GetSupportedRules） 请求应答（Request-Response）	
消息名字	描述
读取支持的规则请求 (GetSupportedRulesRequest)	消息包含需要读取规则列表的 VideoAnalyticsConfigurationToken tt:ReferenceToken ConfigurationToken [1][1]
读取支持的规则响应 （GetSupportedRulesResponse）	响应包含支持的规则 tt: SupportedRules SupportedRules [1][1]
故障代码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	视频分析配置不存在

17.2.4.2 读取规则（GetRules）

以下操作检索当前安装的规则：

表 234：读取规则命令

读取规则（GetRules） 请求应答（Request-Response）	
消息名字	描述
读取规则请求 (GetRulesRequest)	消息包含需要报道的 VideoAnalyticsConfigurationToken 的规则 tt:ReferenceToken ConfigurationToken [1][1]
读取规则响应 （GetRulesResponse）	响应包含指定配置安装的规则列表 tt:Config Rule [0][unbounded]
故障代码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	视频分析配置不存在

17.2.4.3 创建规则（CreateRules）

以下操作是添加规则到一个视频分析配置中。假如所有的规则不能按照请求的来创建，那么设备返回一个错误的消息。

表 235：创建规则命令

创建规则（CreateRules） 请求应答（Request-Response）	
消息名字	描述
创建规则请求 (CreateRulesRequest)	消息指定需要增加规则列表到的 VideoAnalyticsConfigurationToken tt:ReferenceToken ConfigurationToken [1][1] tt:Config Rule [1][unbounded]

创建规则响应（CreateRulesResponse）	这是一个空消息
故障代码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	视频分析配置不存在
env:Sender ter:InvalidArgVal ter:InvalidRule	建议的规则设备不适合这个设备
env:Sender ter:InvalidArgVal ter:RuleAlreadyExistent	同样的规则名称已经存在了配置中
enc:Receiver ter:Action ter:TooManyRules	没有足够的空间来增加规则
env:Receiver ter:Action ter:ConfigurationConflict	设备不能创建规则在以防创建一个相冲突的配置

17.2.4.4 修改规则（ModifyRules）

以下操作用于修改规则，假如所有的规则不能按请求修改，那么设备返回一个错误的提示消息。

表 236：修改规则命令

修改规则（ModifyRules）	请求应答（Request-Response）
消息名字	描述
修改规则请求 (ModifyRulesRequest)	消息包含需要修改规则列表到的 VideoAnalyticsConfigurationToken tt:ReferenceToken ConfigurationToken [1][1] tt:Config Rule[1][unbounded]
修改规则响应 （ModifyRulesResponse）	这是一个空消息
故障代码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	视频分析配置不存在
env:Sender ter:InvalidArgVal ter:InvalidRule	建议的规则设备不适合这个设备
env:Sender ter:InvalidArgs ter:RuleNotExistent	规则名称不存在
enc:Receiver ter:Action ter:TooManyRules	没有足够的空间来增加规则

env:Receiver ter:Action ter:ConflictingConfig	设备不能修改规则以防创建一个相冲突的配置

17.4.4.5 删除规则（DeleteRules）

以下的操作用于删除多个规则，假如所有的规则不能按照请求删除，那么设备会返回一个错误提示消息。

表 237：删除规则命令

删除规则（DeleteRules）	请求应答（Request-Response）
消息名字	描述
删除规则请求 (DeleteRulesRequest)	消息包含需要删除指定规则列表的 VideoAnalyticsConfigurationToken tt:ReferenceToken ConfigurationToken [1][1] xs:string RuleName [1][unbounded]
删除规则响应 （DeleteRulesResponse）	这是一个空消息
故障代码	描述
env:Sender ter:InvalidArgVal ter:NoConfig	视频分析配置不存在
env:Receiver ter:Action ter:ConflictingConfig	不能删除规则在以防创建一个相冲突的配置
env:Sender ter:InvalidArgs ter:RuleNotExistent	规则名称不存在

17.3 分析模块接口

视频分析配置由两个部分组成（见 11.9 章节），第一部分配置视频分析引擎，用于创建场景描述。第二部分配置规则引擎。17.3.1 定义了一个 XML 结构，用于视频分析引擎传递分析模块配置。17.3.2 章节定义了描述特定分析模块配置的语言。17.3.3 定义了分析模块接口请求的操作。假如设备支持一个按照 ONVIF 定义的分析引擎，那么它就能够执行所有的分析模块接口。

17.3.1 分析模块配置

分析模块配置与规则配置相同，详见 17.2.1 章节。下面的例子讲述了一个可能的配置，用于指定的对象跟踪。跟踪装置允许配置关于处理几何图像的最小和最大尺寸。

```
<tt:VideoAnalyticsConfig>
<tt:AnalyticsEngineConfig>
<tt:AnalyticsModule Name="MyObjectTracker" Type="nn:ObjectTracker">
<tt:Parameters>
<tt:SimpleItem Name="MinObjectWidth" Value="0.01"/>
```

```
<tt:SimpleItem Name="MinObjectHeight" Value="0.01"/>
<tt:SimpleItem Name="MaxObjectWidth" Value="0.5"/>
<tt:SimpleItem Name="MaxObjectHeight" Value="0.5"/>
</tt:Parameters>
</tt:AnalyticsModule>
</tt:AnalyticsEngineConfig>
<tt:RuleEngineConfig>
...
</tt:RuleEngineConfig>
</tt:VideoAnalyticsConfig>
```

17.3.2 分析模块描述语言

分析模块描述语言的重复使用，详见 17.2.2，以下讲的是分析模块描述元素代替规则描述元素：

```
<xs:element name="AnalyticsModuleDescription"
type="tt:ConfigDescription"/>
```

与规则类似，分析模块产生的事件，应该列在分析模块描述中。后面的描述符合于上面章节的例子。当场景在模块中变得非常复杂的时候，那么可以创建一个场景拥挤事件，例子如下：

```
<tt:AnalyticsModuleDescription Name="nn:ObjectTracker">
<tt:Parameters>
<tt:SimpleItemDescription Name="MinObjectWidth" Type="xs:float"/>
<tt:SimpleItemDescription Name="MinObjectHeight" Type="xs:float"/>
<tt:SimpleItemDescription Name="MaxObjectWidth" Type="xs:float"/>
<tt:SimpleItemDescription Name="MaxObjectHeight" Type="xs:float"/>
</tt:Parameters>
<tt:MessageDescription>
<tt:Source>
<tt:SimpleItemDescription Name="VideoSourceConfigurationToken"
Type="tt:ReferenceToken"/>
<tt:SimpleItemDescription Name="VideoAnalyticsConfigurationToken"
Type="tt:ReferenceToken"/>
<tt:SimpleItemDescription Name="AnalyticsModule" Type="xs:string"/>
</tt:Source>
<tt:ParentTopic>
tns1:VideoAnalytics/nn:ObjectTracker/SceneTooCrowded
</tt:ParentTopic>
</tt:MessageDescription>
</tt:AnalyticsModuleDescription>
```

17.3.3 分析模块操作

假如设备支持规范定义的规则引擎，那么它就具备以下管理规则的能力，创建/删除/修改命令具有原子性的特点，也就是说所有的操作要么都能被处理，要么都不能被处理。

17.3.3.1 读取支持的分析模块（GetSupportedAnalysticsModule）

通过执行 GetSupportedAnalysticsModule 操作，设备可以给出所有支持的分析模块。它通过分析模块描述语言来返回一系列分析模块，见 17.2.2 章节，此外，它还包含一系列提供概要文件位置的 URL，这些概要文件用于描述在模块分析描述中的类型和元素。假如模块分析描述涉及到 ONVIF 概要文件中的类型或元素，ONVIF 概要文件应该明确地列出。

表 238：读取支持的模块分析命令

读取支持分析模块（GetSupportedAnalyticsModules） 请求应答（Request-Response）	
消息名字	描述
读取支持分析模块请求 (GetSupportedAnalyticsModulesRequest)	消息包含需要列出支持分析模块的 VideoAnalyticsConfigurationToken tt:ReferenceToken ConfigurationToken [1][1]
读取支持分析模块响应 (GetSupportedAnalyticsModulesResponse)	响应包含支持的分析模块 SupportedAnalyticsModules [1][1]
故障代码	描述
env:Sender ter:InvalidArgs ter:NoConfig	视频分析配置不存在

17.3.3.2 读取模块分析（GetAnalyticsModules）

以下操作检索当前按装的分析模块：

表 239：读取分析模块命令

读取分析模块规则（GetAnalyticsModules） 请求应答（Request-Response）	
消息名字	描述
读取分析模块请求 (GetAnalyticsModulesRequest)	这个消息包含需要报道分析模块的 VideoAnalyticsConfigurationToken tt:ReferenceToken ConfigurationToken [1][1]
读取分析模块响应 (GetAnalyticsModulesResponse)	响应包含指定配置安装的分析模块列表 tt:Config AnalyticsModule [0][unbounded]
故障代码	描述
env:Sender ter:InvalidArgs ter:NoConfig	视频分析配置不存在

17.3.3.3 创建分析模块（CreateAnalyticsModules）

接下来的操作是添加分析模块到一个视频分析配置中去，假如所有的分析模块不能按照请求的被创建，那么设备就返回一个错误提示消息。

表 240：创建分析模块命令

创建分析模块规则（CreateAnalyticsModules） 请求应答（Request-Response）	
消息名字	描述
创建分析模块请求 (CreateAnalyticsModulesRequest)	这个消息包含需要创建分析模块的 VideoAnalyticsConfigurationToken tt:ReferenceToken ConfigurationToken [1][1] tt:Config AnalyticsModule [1][unbounded]
创建分析模块响应 (CreateAnalyticsModulesResponse)	空消息 tt:Config AnalyticsModule [0][unbounded]
故障代码	描述
env:Sender ter:InvalidArgs ter:NoConfig	视频分析配置不存在
env:Sender ter:InvalidArgs ter:NameAlreadyExistent	配置中已存在相同名称的分析模块
enc:Receiver ter:Action ter:TooManyModules	设备中没有足够的空间来添加配置的分析模块
env:Receiver ter:Action ter:ConfigurationConflict	设备不能创建一个相冲突的分析模块给配置文件
env:Sender ter:InvalidArgVal ter:InvalidModule	建议的模块配置不适合设备

17.3.3.4 修改分析模块（ModifyAnalyticsModules）

下来的操作是修改多个分析模块，假如所有的分析模块不能按照请求的被修改，那么设备就返回一个错误提示消息。

表 241：修改分析模块命令

修改分析模块规则（ModifyAnalyticsModules） 请求应答（Request-Response）	
消息名字	描述
修改分析模块请求 (ModifyAnalyticsModulesRequest)	这个消息包含需要修改分析模块的 VideoAnalyticsConfigurationToken tt:ReferenceToken ConfigurationToken [1][1] tt:Config AnalyticsModule [1][unbounded]
修改分析模块响应 (ModifyAnalyticsModulesResponse)	空消息 tt:Config AnalyticsModule [0][unbounded]
故障代码	描述
env:Sender ter:InvalidArgs ter:NoConfig	视频分析配置不存在
env:Sender	请求的分析模块名称不存在

ter:InvalidArgs ter:NameNotExistent	
enc:Receiver ter:Action ter:TooManyModules	设备中没有足够的空间来添加分析模块到配置中
env:Receiver ter:Action ter:ConfigurationConflict	不能修改配置文件以防创建相冲突的配置
env:Sender ter:InvalidArgVal ter:InvalidModule	建议的模块配置不适合设备

17.3.3.5 删除分析模块（DeleteAnalyticsModules）

下来的操作是删除多个分析模块，假如所有的分析模块不能按照请求的被删除，那么设备就返回一个错误提示消息。

表 242：删除分析模块命令

删除分析模块规则（DeleteAnalyticsModules）	请求应答（Request-Response）
消息名字	描述
删除分析模块请求 (DeleteAnalyticsModulesRequest)	这个消息包含需要删除分析模块的视频分析配置标记 VideoAnalyticsConfigurationToken tt:ReferenceToken ConfigurationToken [1][1] xs:string AnalyticsModuleName [1][unbounded]
删除分析模块响应 (DeleteAnalyticsModulesResponse)	空消息 tt:Config AnalyticsModule [0][unbounded]
故障代码	描述
env:Sender ter:InvalidArgs ter:NoConfig	视频分析配置不存在
env:Receiver ter:Action ter:ConfigurationConflict	设备不能删除分析模块以防创建一个相冲突的配置
env:Sender ter:InvalidArgs ter:NameNotExistent	请求的分析模块名称不存在

17.4 服务错误码

表 243 列的是分析服务错误码，每个命令也能够生成一种类错误，详见表 6。
特定的错误相当于一个类错误的子代码，见 5.11.2.1.父类子代码是上一级错误码的子代码，特定的错误子代码是最底层的错误子代码。

表 243：服务错误码

错误码	父类子错误码码 子错误码	错误原因	描述
env:接收器（Receiver）	ter: Action	没有更多可用空间	设备没有足够的空间来添加规则到配置中
	ter:TooManyRules		
env:接收器（Receiver）	ter:Action	没有更多可用空间	设备没有足够的空间来添加分析模块到配置中
	ter:TooManyModules		
env:接收器（Receiver）	ter: Action	当用新的设置发生冲突	新的设置导致了配置冲突
	ter:ConfigurationConflict		
env: 发射机（Sender）	ter: InvalidArgVal	没有这样的配置	请求的视频分析配置不存在
	ter: NoConfig		
env: 发射机（Sender）	ter: InvalidArgVal	规则无效	设置的规则配置不适合
	ter: InvalidRule		
env: 发射机（Sender）	ter: InvalidArgVal	模块无效	建议的分析模块配置不适合设备
	ter: InvalidModule		
env: 发射机（Sender）	ter: InvalidArgVal	规则存在	请求的规则名称已经存在配置中
	ter:RuleAlreadyExistent		
env:发射机（Sender）	ter: InvalidArgs	规则不存在	规则名称不存在
	ter:RuleNotExistent		
env:发射机（Sender）	ter: InvalidArgs	名称存在	相同的分析模块名称已经存在配置中
	ter: NameAlreadyExistent		
env:发射机（Sender）	ter: InvalidArgs	名称不存在	请求的分析模块名

	ter: NameNotExistent		称不存在

18 分析设备

分析设备服务必须用于独立的分析设备，这些独立的分析设备能够对流媒体或者增强媒体流的元数据执行评估。同时分析设备服务同样适用于其他实体。分析设备可以一次对多个的流媒体或者增强流媒体元数据进行评估。

分析设备服务接收的流媒体或者增强流媒体的元数据，这些数据要么来自直接产生要么来自存储设备。如果分析的是一个没有压缩的数据，那么分析设备应该具备解码的能力。加强的流媒体元数据能够描述任何包含媒体数据的流媒体以及指定的元数据。

客户端使用分析设备服务来配置独立分析设备属性和功能或者配置其他一些带分析功能的实体的属性和功能。

独立的分析设备不提供反向通道功能。

分析设备服务依赖于对其它设备上的接收数据服务，这接收服务通过 ReceiverTokens 来识别接收对象来完成接收数据。在接受 RTSP 流时，必须提供一种机制来分配不同的路径给相应 AnalyticsEngine。

当正在执行分析时，改变参数（例如变化摄像头参数）可能会影响分析结果。因此，输入参数的变化反映在 AnalyticsEngineInput 结构上。

18.1 概述

在配置一个分析设备服务的核心元素是 AnalyticsEngineControl 。它包含必要令牌和对服务的描述以及对 AnalyticsEngineControl 进行激活/暂停。

一个 AnalyticsEngine 可能是一个算法或者一个完整的应用程序，例如丢包。几个参数集（ VideoAnalyticsConfiguration ）可以并行在 AnalyticsEngine 存在，并且允许它们之间切换如：例如：白天和黑夜的配置。此外，提供一个结构体来对这特别的 AnalyticsEngine 进行描述。

为了使 AnalyticsEngine 适应不同的输入数据，在 AnalyticsEngineInput 元素中，必须提供对输入进 AnalyticsEngine 数据元素提供描述

所有结构体在启动 NVA 实体后必须至少存在一次，以及这些结构体能够在合适的位置填充默认的值。

18.2 分析引擎输入

AnalyticsEngineInput 结构体描述了输入的视频以及元数据，这些数据是用来提供给特别

的 AnalyticsEngine。如果超过一种输入源，那么必须对每一个输入源提供一个 AnalyticsEngineInput 元素。

SourceIdentification ：标识输入是来自哪个源（如区别相机集群，特别是摄像头和正在使用一类）
VideoSource ：视频源的信息，特别是关于正使用的压缩参数
MetadataInput：描述提供将要被分析元数据的源。

18.2.1获取分析引擎输入

这个操作列举了所有对设备可用的分析引擎的输入。分析设备服务应该支持列举的可用的分析引擎输入。这个清单是对 GetAnalyticsEngineInputs 命令的应答。

表 244：GetAnalyticsEngineInputs 命令

GetAnalyticsEngineInputs	请求与应答
信息名称	功能描述
GetAnalyticsEngineInputsRequest	请求信息是一条空信息
GetAnalyticsEngineInputsResponse	应答信息包含一个可用分析引擎输入清单： tt:AnalyticsEngineInputConfiguration[1][unbounded]
错误代码	原因分析
	没有与此命令相关的错误代码

18.2.2获取分析引擎的输入

如果分析引擎的输入配置的令牌已知，那么可以通过 GetAnalyticsEngineInput 命令取得输入配置。一个分析设备服务应该支持通过 GetAnalyticsEngineInput 命令来获取分析引擎输入配置清单。

表：245：GetAnalyticsEngineInput 命令

GetAnalyticsEngineInput	请求与应答
信息名称	功能描述
GetAnalyticsEngineInputRequest	请求信息包含一个已经存在分析引擎输入配置的令牌。 tt:ReferenceToken ConfigurationToken [1][1]
GetAnalyticsEngineInputResponse	应答信息包含请求的分析引擎输入配置： tt:AnalyticsEngineInput Configuration [1][1]
错误代码	原因分析

env:Sender ter:InvalidArgVal ter:NoConfig	请求信息中令牌指定的配置不存在。

18.2.3设置分析引擎的输入

这个命令可以改变分析引擎输入配置。一个分析设备服务应该支持通过 SetAnalyticsEngineInput 命令来修改分析引擎输入配置。

表 246：SetAnalyticsEngineInput 命令

SetAnalyticsEngineInput	请求与应答
信息名称	功能描述
SetAnalyticsEngineInput - Request	请求信息应该包含一个新的配置 这 ForcePersistence 元素决定了这改变的配置是否在重启后应该被存储以及保留。如果为真，改变的配置应该被保留；如果是假，那么在重启后恢复先前配置。 tt:AnalyticsEngineInput Configuration[1][1] xs:boolean ForcePersistence [1][1]
SetAnalyticsEngineInputResponse	应答信息是一个空信息
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:invalidConfig	对设备设置不可能的配置。
env:Sender ter:InvalidArgVal ter:NoConfig	请求信息中令牌指定的配置不存在

18.2.4 创建分析引擎输入

通过这个命令可以产生分析引擎输入配置。一个分析设备服务应该支持通过这个命令来产生分析引擎输入配置。

表 247：CreateAnalyticsEngineInputs 命令

CreateAnalyticsEngineInputs	请求与应答
信息名称	功能描述
CreateAnalyticsEngineInputsRequest	请求信息中配置应该是一个新的配置 这 ForcePersistence 元素决定了这改变的配置是否在重启后应该被存储以及保留。如果为真，改变的配置应该被保留；如果是假，那么在重启后恢复先前配置。 tt:AnalyticsEngineInput Configuration[1][unbounded] xs:boolean ForcePersistence [1][unbounded]

CreateAnalyticsEngineInputsResponse	应答信息包含：创建的引擎配置以及令牌 tt:AnalyticsEngineInputConfiguration[1][unbounded]
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:invalidConfig	对设备设置不可能的配置。
env:Receiver ter:Action ter:MaxAnalyticsEngineInput	已经达到分析引擎支持对象的最大数量

18.2.5删除分析引擎输入

通过 DeleteAnalyticsEngineInputs 命令能够删除分析引擎输入配置。一个分析设备服务应该支持通过这个命令来删除分析引擎输入配置。

表 248：DeleteAnalyticsEngineInputs 命令

DeleteAnalyticsEngineInputs	请求与应答
信息名称	功能描述
DeleteAnalyticsEngineInputsRequest	请求信息包含 识别将要删除的分析引擎输入的配置令牌。 tt:ReferenceTokenConfigurationToken[1][unbounded
DeleteAnalyticsEngineInputsResponse	应答信息是一条空信息
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:NoAnalyticsEngineInput	与请求信息中令牌指定的配置信息不存在 。
env:Sender ter:Action ter:CannotDeleteEngineInput	不能够删除请求中指定的分析引擎输入

18.3 视频分析配置

18.3.1 获取视频分析配置

如果视频分析配置的令牌已知，通过 GetVideoAnalyticsConfiguration 命令可以获取视频分析配置。一个分析设备服务应该支持通过 GetVideoAnalyticsConfiguration 命令来获取视频分析配置的清单。所有相应的视频分析配置的令牌可在 AnalyticsEngine 配置中找到。

表 249：GetVideoAnalyticsConfiguration 命令

GetVideoAnalyticsConfiguration	请求与应答

信息名称	功能描述
GetVideoAnalyticsConfigurationRequest	请求信息包含一个存在的视频分析配置的令牌 tt:ReferenceToken ConfigurationToken [1][1]
GetVideoAnalyticsConfigurationResponse	应答信息包含请求的视频分析配置。 tt:VideoAnalyticsConfigurationConfiguration[1]1
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:NoConfig	这请求信息中令牌指定的配置不存在。

18.3.2 设置视频分析配置

这个命令可以改变视频分析配置。一个分析设备服务应该支持通过这个命令来修改分析引擎配置。如果分析设备服务收到这 SetVideoAnalyticsConfiguration 命令。这设置的配置在实际运用中也能受到影响。

表 250：SetVideoAnalyticsConfiguration 命令

SetVideoAnalyticsConfiguration	请求与应答
信息名称	功能描述
SetVideoAnalyticsConfiguration – Request	配置应该是一个新的配置。 这 ForcePersistence 元素决定了这改变的配置应是否在重启以后被存储。如果为真，改变的配置应该被保存。如果为假，恢复重启之前配置 tt:VideoAnalyticsConfigurationConfiguration[1][1 xs:boolean ForcePersistence [1][1]
SetVideoAnalyticsConfigurationResponse	应答信息是一条空信息
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:invalidConfig	设置不可能的配置
env:Sender ter:InvalidArgVal ter:NoConfig	请求信息中令牌指示的配置不存在。

18.4 分析引擎

通过这定义的命令返回一个结构体，这个结构体包含一个对特殊引擎可用 VideoAnalyticsConfiguration 的清单，还包含对每个 AnalyticsEngine 配置的适合 AnalyticsEngineInputInfo 元素。

VideoAnalyticsConfiguration：描述分析引擎可能的配置。

AnalyticsEngineInputInfo：分析引擎输入要求的信息。

18.4.1获取分析引擎

这个操作列举了设备所有可用的分析引擎。分析设备服务应当支持通过 GetAnalyticsEngines 命令来获取的可用分析引擎清单。

表 251：GetAnalyticsEngines 命令

GetAnalyticsEngines	请求与应答
信息名称	功能描述
GetAnalyticsEnginesRequest	请求信息是一条空信息
GetAnalyticsEnginesResponse	应答信息包含：一个描述可用分析引擎结构体清单。 tt:AnalyticsEngine Configuration [1][unbounded]
错误代码	原因分析
	没有与这条命令相关的错误代码

18.4.2 获取分析引擎

如果分析引擎的令牌已知，通过这条 GetAnalyticsEngine 命令能够获取分析引擎。一个分析设备服务应支持通过 GetAnalyticsEngine 命令来获取分析引擎配置清单。

表 252：GetAnalyticsEngine 命令

GetAnalyticsEngine	请求与应答
信息名称	功能描述
GetAnalyticsEngineRequest	请求信息包含一个已经存在的分析引擎的令牌。 tt:ReferenceToken ConfigurationToken [1][1]
GetAnalyticsEngineResponse	应答信息包含请求的 AnalyticsEngine 配置 tt:AnalyticsEngine Configuration [1][1]
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:NoConfig	请求信息中令牌指定的配置不存在

18.5 分析引擎控制

这 AnalyticsEngineControl 结构体应该被用来通过下面的命令进行实际控制。

名称 ：友好的描述
EngineToken ：被控制的分析引擎令牌。

EngineConfigToken：有效分析引擎配置的令牌（包含视频分析配置的令牌）。

InputToken：使用的输入配置令牌

ReceiverToken：提供媒体输入数据的接收令牌。接收令牌的顺序应该准确的对 InputToken 进行准确匹配

Multicast：用来配置广播以及用于控制广播元数据流参数

Subscription：对受控引擎起反应的描述

Mode：表明实际控制分析的状态（"Idle" or "Active）。

18.5.1 GetAnalyticsEngineControls

这个操作列举了设备所有可用的分析引擎。这分析设备服务应支持通过 GetAnalyticsEngineControls 命令来获取可用分析引擎的清单。

表 253：GetAnalyticsEngineControls 命令

GetAnalyticsEngine	请求与应答
信息名称	功能描述
GetAnalyticsEngineControlsRequest	请求信息是一个空信息
GetAnalyticsEngineControlsResponse	应答信息包含一个描述可用分析引擎控制的结构体清单 tt:AnalyticsEngineControl AnalyticsEngineControls[1][unbounded
错误代码	原因分析
	没有与此命令相关的错误代码

18.5.2 获取分析引擎控制

如果引擎控制的令牌可知，那么通过 GetAnalyticsEngineControl 命令来获取对分析引擎的控制。一个分析设备服务应支持通过 GetAnalyticsEngineControl 命令来获取分析引擎配置列表。

表 254：GetAnalyticsEngineControl 命令

GetAnalyticsEngineControl	请求与应答
信息名称	功能描述
GetAnalyticsEngineControlRequest	请求信息是包含存在的 AnalyticsEngineControl 令牌 tt:ReferenceToken ConfigurationToken [1][1]
GetAnalyticsEngineControlResponse	应答信息包含请求的 AnalyticsEngineControl 配置 tt:AnalyticsEngineControlConfiguration[1][1]
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:NoConfig	与请求中令牌指定的配置不存在

18.5.3设置分析引擎控制

这个命令可以改变 AnalyticsEngineControl 配置。一个分析设备服务应支持通过这个命令来设置对分析引擎控制配置。

表 255：SetAnalyticsEngineControl 命令

SetAnalyticsEngineControl	请求与应答
信息名称	功能描述
SetAnalyticsEngineControlRequest	配置应该是新的配置 这 ForcePersistence 元素决定改变的配置是否在重启后被存储以及保持，如果为真，改变的配置被保存，如果为假，改变的配置恢复启动前配置 tt:AnalyticsEngineControl Configuration[1][1] xs:boolean ForcePersistence [1][1]
SetAnalyticsEngineControlResponse	应答信息是一条空信息
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:invalidConfig	设置的配置不可能实现
env:Sender ter:InvalidArgVal ter:NoConfig	请求中令牌指定配置不存在

18.5.4 CreateAnalyticsEngineControl

CreateAnalyticsEngineControl 创建一个新的控制对象。方式应该被设置为空闲状态。应该通过 SetAnalyticsEngineControl 命令来改变“active”方式。分析设备服务应该支持通过这个命令来创建控制对象。

表 256：CreateAnalyticsEngineControl 命令

CreateAnalyticsEngineControl	请求与应答
信息名称	功能描述
CreateAnalyticsEngineControlRequest	配置应该是新的配置 tt:AnalyticsEngineControlConfiguration[1][1]
CreateAnalyticsEngineControlResponse	应答信息请求的配置以及相关的令牌： tt:AnalyticsEngineControlConfiguration[1][1
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:AnalyticsEngineControlExists	请求信息中的令牌指定的分析引擎控制已经存在
env:Receiver ter:Action ter:MaxAnalyticsEngineControl	已经达到 AnalyticsEngineControl 支持的最大的对象数
env:Sender ter:InvalidArgVal ter:invalidConfig	设置的配置不可能

18.5.5删除分析引擎控制

DeleteAnalyticsEngineControl 应该删除一个控制对象。一个分析设备服务应当支持通过这个命令删除控制对象。

表 257：DeleteAnalyticsEngineControl 命令

DeleteAnalyticsEngineControl	请求与应答
信息名称	功能描述
DeleteAnalyticsEngineControlRequest	请求信息包含将要删除控制对象的令牌 tt:ReferenceToken ConfigurationToken [1][1]
DeleteAnalyticsEngineControlResponse	应答信息是一条空信息
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:NoAnalyticsEngineControl	请求中令牌指定的控制对象不存在
env:Receiver ter:Action ter:MaxAnalyticsEngineControl	指定分析引擎控制不能够被删除

18.6 获取分析状态

GetAnalyticsState 返回与分析引擎控制对象相关的状态信息。分析状态信息的结构体是可以扩展的。这扩展应该被用来表达额外的信息。例如：一个分析引擎应该有不同的分析算法组成，这些算法的状态信息应该被提供。这 AnalyticsStateInformation 状态元素总是包含所有子结构的聚集状态

一个分析设备服务应该支持通过这个命令来提供状态信息。

当状态信息被请求时，这 AnalyticsEngineControl 应该通过 ConfigurationToken 来表标识。

状态应该包含所有子结构的聚集状态，如果状态是错误的，这错误可能用一个生效的定义值来填充。一个设备运用以下规则来计算聚集状态：

Idle	所有子结构体状态都是空闲状态
PartiallyActive	至少有一个子结构状态是 AVTIC，所有其他子结构必须是空闲
Active	所有子结构是 ACTIV 状态
Error	至少有一个子结构状态是错误的

错误，如果出现，应该有一个完整定义的字符串描述这个错误

表 258：GetAnalyticsState

GetAnalyticsState	请求与应答
信息名称	功能描述
GetAnalyticsStateRequest	请求信息包含 AnalyticsEngineControl 配置令牌： tt:ReferenceToken ConfigurationToken [1][1]
GetAnalyticsStateResponse	应答信息是 AnalyticsEngineControl.状态信息： tt:AnalyicsStateInformation State[1][1]
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:NoAnalyticsEngineControl	这配置的令牌与存在 AnalyticsEngineControl.不相关

18.7 输出流配置

一个分析设备服务提供一个像在“实时流”中定义的实时流接口。令牌识别代替了在媒体类别中使用的预置文件（Profile），在 AnalyticsEngineControl 中令牌识别适用于分析设备服务。

18.7.1 请求流的URL

这个操作可以用来请求一个 URL；这个 URL 能够被用来初始化一个实时流，这个实时流使用 RTSP 协议作为控制协议的。只有在响应中指定的 URL 或者分析引擎控制被再配置时，

URI 是有效的。分析设备服务应该支持通过 GetAnalyticsDeviceStreamUri 命令来检索一个 URL 流，这个 URL 流是与分析引擎控制相关的.

表 259：GetAnalyticsDeviceStreamUri 命令

GetAnalyticsDeviceStreamUri	请求与应答
信息名称	功能描述
GetAnalyticsDeviceStreamUriRequest	请求信息包含 StreamSetup 元素包含两部分：流的类型定义 t（当单播或者广播被请求时） 按指定的线路或者定义在不同网络的协议来传输： AnalyticsEngineControlToken 元素应该标明分析引擎控制使用 tt:StreamSetup StreamSetup [1][1] tt:ReferenceTokenAnalyticsEngineControlToken [1][1]
GetAnalyticsDeviceStreamUriResponse	应答信息是请求的 URL 信息： xs:anyURI Uri [1][1]
错误代码	原因分析
env:Sender ter:InvalidArgVal ter:NoAnalyticsEngineControl	用令牌指定的配置不存在
env:Sender ter:InvalidArgVal ter:InvalidStreamSetup	不支持协议的流类型或者流 StreamSetup 的传输
env:Sender ter:OperationProhibited ter:StreamConflict	协议的流类型或者在流设置的传输部分与其它流相冲突

19 录制控制

19.1 介绍

录像服务，使客户端能够管理录像并设置从数据源到录像带的数据的转移。录制管理，包括创建和删除录制，以及锁定、解锁录制和删除录制数据。

录制工作从录像源传输数据到录像带。录像的来源可以是接收器服务创建的接收对象，或者是一个本地设备上的编码数据的媒体文件。媒体文件可以用来作为嵌入式存储摄像头的源。

在本规范中使用录制来存储一组音频、视频和元数据的轨道集。录像带可容纳任意数量的轨道集。轨道被看做是在特定的时间记录数据的无限的时间线。

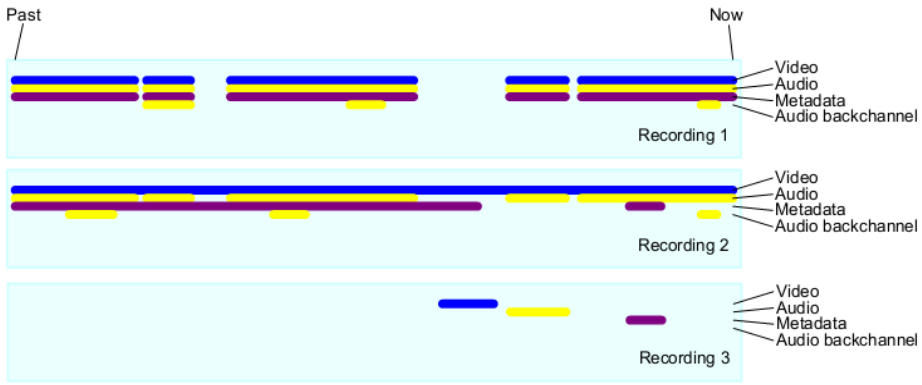


图 24：录像的例子

该图显示了三个录制集，每一个有视频，元数据和两个音轨。在这里，第二个音轨用于存储音频反向通道。

录制至少应当是有三个轨道，分别给音频、视频和元数据。录像服务的某些实现方式可能支持各种类型，多种轨道集。

为了保存录制的数据，客户端首先创建一个录像，保证录像有必要的轨道。然后在客户端创建一个录像的工作，从一个或多个数据源提取数据然后把数据存储到录像轨道。

客户可以设立多个录制任务，且都录制到相同的录像带。如果多个录像工作同时活动，该设备使用优先计划去选择在轨道之间定义的录像工作。客户可随时更改录像的工作模式，从而可以实现报警录像或手动录像等功能。

录像工作依赖于接收器服务通过接收对象的接收令牌来确认从其他设备接收数据。

客户端使用单一录像工作的接收器对象的情况下，录制服务可以自动创建和自动删除接收对象。自动生成与自动创建接收信号标志在录像作业配置结构中。当没有录像工作使用时，接收器对象创造的这种方式将被自动删除。一个自动创建的接收器对象把它所有领域设置为空值。客户应在它建立录像工作后设定接收对象。

ONVIF 的录像观点是一个合乎逻辑的、独立的方式，录像存储在物理磁盘上。例如，一些摄像头装置在文件系统中为每个报警的发生创建一个不同的文件来实现警报录像。然而每个文件可以被表示为一个不同的 ONVIF 录像，该标准模型的目的是把所有的文件汇总到一个单一的录制。然后通过 FindEvents 方法搜索“datapresent”事件搜索服务，用户可以设置视频什么时候开始录制、什么时候停止录制。

如果元数据被录制，则元数据也可以容纳所有的事件所产生的数据源（见 15 节和 11.10 节的事件处理数据的配置对象）。此外，设备在理论上也录制 ONVIF 定义的历史事件（见搜索服务的录制事件描述），例如开始和结束录制数据范围。设备在理论上也可以录制供应商特定的历史事件。设备产生的事件不插入现有的录制的元数据轨道。搜索服务中的 FindEvents 方法中可以找到的所有录制事件。许多设备实现自动删除最古老的录制数据存储，为新录制释放空间。锁存提供一个允许用户选择数据范围的机制。一系列被锁存的数据是不会自动被删除的。支持锁存是保留给未来的规范版本。

19.2 一般要求

所有对象创建的录制应当持续－即他们继续存在动力循环。同样，所有的配置数据的对象应持续。

19.3 数据结构

19.3.1 录制设置

录制配置结构通过 CreateRecordings 和 Get/SetRecordingConfiguration 设置录制。

MaximumRetentionTime 指定数据在录制的任何轨道存放的最长时间。设备应删除超过了最大限度保留时间的所有数据。这些数据将无法访问了。如果 maximumretentionperiod 设置为 0，设备不应限制存储数据的保留时间，除由资源约束。然而无论 maximumretentiontime 是否存在，设备可以自动删除录制为新录制释放存储空间。

设备仅使用此结构的这个功能。因此，它只是存储信息，并将存储的信息通过 GetRecordingConfiguration 和 GetRecordingInformation 方式返回（见 20.5）。

19.3.2 轨迹设置

轨道配置结构使用 CreateTrack 和 Get/SetTrackConfiguration 配置轨道。

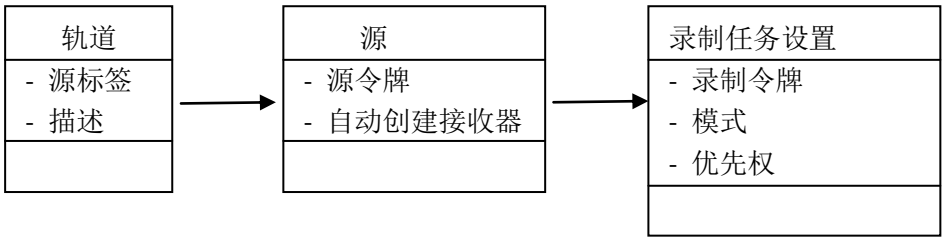
轨道设置包含以下领域：

TrackType 定义轨道的数据类型。应当等于字符串“视频”,“音频”或“元数据”。轨道只能保存这些类型的数据。

设备仅使用此结构的这个功能。因此，它只是存储信息，并将存储的信息通过 GetTrackConfiguration 和 GetRecordingInformation 方式返回（见 20.5）。

19.3.3 录制任务设置

录制任务设置结构保存录制任务的设置。录制任务设置可以通过以下图例呈现：



录制任务设置包含以下领域：

RecordingToken：标识录制这项工作应存储接收到的数据。

Mode：工作模式。如果是闲置，没有什么会发生。如果它是有效的，该装置应设法从接收器获取数据。客户端应当使用 GetRecordingJobState 确定数据传输是否是真的发生。有效模式只有“闲置”和“活动”。

Priority：优先权，这是一个正数。如果有多个录制任务在同一轨道上存储数据，该设备将只存储最高优先权的录制任务的数据。优先指定各个录制任务，但该设备会鉴定单独优先哪个轨道。如果录制任务具有相同的优先级，设备将录制最新的录制任务对应的数据。

0 表示最低优先级。数字越大表示更高的优先权。

SourceToken：这是一个数据源的参考。源类型在 SourceToken 结构中被确定属性类型。
如果类型是 http://www.onvif.org/ver10/schema/receiver，令牌是一个接收器参考。在这种情况下该装置应通过网络接收数据。
如果类型是 http://www.onvif.org/ver10/schema/profile，令牌标识媒体文件，指示设备在本地设备从文件获得的数据。

如果 SourceToken 省略，AutoCreateRecevier 应是 true。

AutoCreateReceiver: 如果此字段是真且 SourceToken 省略，则该装置会创建一个接收器对象（通过接收器服务）和分配接收器参考到 SourceToken。当从设备检索录制任务设置时，AutoCreateReceiver 永远为真。

SourceTag: 如果收到的 RTSP 流包含多个相同类型的轨道，Sourcetag 会区分这些轨道。

Destination: 目的地是该设备应存储接收到的数据的轨道的 TrackToken。

19.4 创建录制

CreateRecording 创建一个新的录制。新的录制必须分别创建一个视频、一个音频和一个元数据轨道。

此方式是可选的。如果 Recording/DynamicRecordings 是真则必须有效。

表 260：CreateRecording 命令

CreateRecording	请求-响应
消息名称	描述
CreateRecordingRequest	包含录制的初始设置。 tt:RecordingConfiguration RecordingConfiguration[1][1]
CreateRecordingResponse	返回创建录制的参考。 tt:RecordingReference RecordingToken[1][1]
故障码	描述
env:Receiver ter:Action ter:MaxRecordings	设备不能创建一个新的录制，已达到支持的最大录制数量。
env:Sender ter:InvalidArgVal ter:BadConfiguration	录制设置无效。
env:Receiver	此优化方式不实现。

ter:ActionNotSupported ter:NotImplemented	

成功完成时，创建录制应已创建三条轨道配置如下：

轨道令牌	轨道类型
VIDEO001	视频
AUDIO001	音频
META001	元数据

所有的轨道配置的最大限度的保留时间应设置为 0（无限），描述设置为空字符串。

19.5 删除录制

删除录制将删除录制对象。当一个录制被删除，设备将删除录制的所有轨道，并删除所有录制任务录制到的录像。每个删除的录制任务，设备也将删除所有使用录制任务的配置结构的 AutoCreateReceiver 自动创建的录制任务的接收器对象，并将不用于任何其他录制任务。此方式是可选的。如果 Recording/DynamicRecordings 是真则必须有效。

表 261：DeleteRecording 命令

DeleteRecording	请求-响应
消息名称	描述
DeleteRecordingRequest	标识应删除的录制。 tt:RecordingReference　RecordingToken[1][1]
DeleteRecordingResponse	空消息。
故障码	描述
env:Sender ter:InvalidArgVal ter:NoRecording	录制令牌不提供现有录制的参考。
env:Receiver ter: ActionNotSupported ter:NotImplemented	设备不能删除录像。
env:Receiver ter:Action ter:CannotDelete	特定的录像不能被删除。

19.6 获取录制集

GetRecordings 应当返回一个设备所有的录制的说明书。说明书包括所有录制各自的轨道清单。

表 262：GetRecordings 命令

GetRecordings	请求-响应
消息名称	描述
GetRecordingsRequest	空消息。

GetRecordingsResponse	RecordingItem 识别录制及其当前组态。 tt:GetRecordingsResponseItem RecordingItem[0][unbounded]
故障码	描述
没有具体的故障码。	

19.7 设置录制配置

SetRecordingConfiguration 改变录制的设置。

表 263：SetRecordingConfiguration 命令

SetRecordingConfiguration	请求-响应
消息名称	描述
SetRecordingConfigurationRequest	RecordingToken 判定录像是否改变。 RecordingConfiguration 应是录像的新设置。 tt:RecordingReference RecordingToken[1][1] tt:RecordingConfiguration RecordingConfiguration[1][1]
SetRecordingConfigurationResponse	空消息。
故障码	描述
env:Sender ter:InvalidArgVal ter: BadConfiguration	设置无效。
env:Sender ter:InvalidArgVal ter:NoRecording	录制令牌不提供现有录像的参考。

19.8 获取录制配置

GetRecordingConfiguration 检索录像的录制设置。

表 264：GetRecordingConfiguration 命令

GetRecordingConfiguration	请求-响应
消息名称	描述
GetRecordingConfigurationRequest	RecordingToken 判定录像的什么设置被检索。 tt:RecordingReference RecordingToken[1][1]
GetRecordingConfigurationResponse	RecordingConfiguration 应是当前录像的设置 tt:RecordingConfiguration RecordingConfiguration[1][1]
故障码	描述
env:Sender ter:InvalidArgVal ter:NoRecording	录制令牌不提供现有录像的参考。

19.9 创建轨道

此方式创建一个新的录制轨道。
此方式是可选的，如果 Recording/DynamicTracks 是真则必须有效。

表 265：CreateTrack 命令

CreateTrack	请求-响应
消息名称	描述
CreateTrackRequest	RecordingToken 判定录像应增加哪一个轨道。 TrackConfiguration 提供设置的新轨道。 tt:RecordingReference RecordingToken[1][1] tt:TrackConfiguration TrackConfiguration [1][1]
CreateTrackResponse	TrackToken 判定新建立的轨道。TrackToken 是录像唯一属于新轨道的。 tt:TrackReference TrackToken[1][1]
故障码	描述
env:Sender ter:InvalidArgVal ter:NoRecording	录制令牌不提供现有录像的参考。
env:Receiver ter:Action ter:MaxTracks	新轨道不能被建立，因为该录像已达到设备支持的最大轨道数。
env:Sender ter:InvalidArgVal ter: BadConfiguration	设置无效。
env:Receiver ter: ActionNotSupported ter:NotImplemented	不能实现该方法。

一个 TrackToken 本身并不是唯一地标识一个特定的轨道。轨道内不同的录制，可能具有相同的轨道令牌。

19.10 删除轨道

DeleteTrack 输出录像的轨道。所有在轨道上的数据都会被删除。

此方式可选的，如果 Recording/DynamicTracks 是真则必须有效。

表 266：DeleteTrack 命令

DeleteTrack	请求-响应
消息名称	描述
DeleteTrackRequest	RecordingToken 判定录像应删除哪一个轨道。 tt:RecordingReference RecordingToken[1][1] tt:TrackConfiguration TrackToken [1][1]
DeleteTrackResponse	空消息。
故障码	描述
env:Receiver ter: ActionNotSupported ter:NotImplemented	设备不执行 DeleteTrack 方式。

env:Sender ter:InvalidArgVal ter:NoTrack	轨道令牌不提供现有轨道的参考。
env:Sender ter:InvalidArgVal ter:NoRecording	录制令牌不提供现有录制的参考。
env:Receiver ter:Action ter: CannotDelete	指定轨道不能删除。

19.11 获取轨道配置

GetTrackConfiguration 检索指定轨道的设置。

表 267：GetTrackConfiguration 命令

GetTrackConfiguration	请求-响应
消息名称	描述
GetTrackConfigurationRequest	RecordingToken 和 TrackToken 判定从哪里获得录制轨道的设置。 tt:RecordingReference RecordingToken[1][1] tt: TrackReference TrackToken [1][1]
GetTrackConfigurationResponse	tt:TrackConfiguration TrackConfiguration [1][1]
故障码	描述
env:Sender ter:InvalidArgVal ter:NoTrack	轨道令牌不提供现有轨道的参考。
env:Sender ter:InvalidArgVal ter:NoRecording	录制令牌不提供现有录制的参考。

19.12 设置轨道配置

SetTrackConfiguration 将改变轨道的设置。

表 268：SetTrackConfiguration 命令

SetTrackConfiguration	请求-响应
消息名称	描述
SetTrackConfigurationRequest	RecordingToken 和 TrackToken 判定轨道应设置哪一个轨道。TrackConfiguration 是轨道的新设置。 tt:RecordingReference RecordingToken[1][1] tt:TrackReference TrackToken[1][1] tt:TrackConfiguration TrackConfiguration [1][1]
SetTrackConfigurationResponse	空消息。
故障码	描述
env:Sender	轨道令牌不提供现有轨道的参考。

ter:InvalidArgVal ter:NoTrack	
env:Sender ter:InvalidArgVal ter:NoRecording	录制令牌不提供现有录制的参考。
env:Sender ter:InvalidArgVal ter: BadConfiguration	设置无效。

19.13 创建录制任务

CreateRecordingJob 将建立一个新的录制任务。

表 269：CreateRecordingJob 命令

CreateRecordingJob	请求-响应
消息名称	描述
CreateRecordingJobRequest	JobConfiguration 将保持新录制任务的设置。 tt:RecordingJobConfiguration JobConfiguration [1][1]
CreateRecordingJobResponse	JobToken 将鉴定建立的录制任务。JobConfiguration 结构是设备使用的设置。JobConfiguration 将有别于 CreateRecordingJob。 tt:RecordingJobReference JobToken[1][1] tt:RecordingJobConfiguration JobConfiguration [1][1]
故障码	描述
env:Receiver ter:Action ter:MaxRecordingJobs	已达到设备能掌握的最大录制任务。
env:Sender ter:InvalidArgVal ter: BadConfiguration	JobConfiguration 内容无效。
env:Receiver ter:Action ter:MaxReceivers	AutoCreateReceivers 标志为真，如果接收服务不能建立一个新的接收，此错误将返回。

从 CreateRecordingJob 返回的任务配置应和任务配置传递到 CreateRecordingJob 的相同，除了接收器令牌和自动创建接收器。在返回的结构中，接收器令牌应当有效且应忽略自动创建接收器。

19.14 删除录制任务

DeleteRecordingJob 将删除一个录制任务。同时它将删除所有接收对象和录制任务使用 AutoCreateReceiver 自动创建的录制任务配置结构，且其他任何录制任务将不再使用。

表 270：DeleteRecordingJob 命令

DeleteRecordingJob	请求-响应

消息名称	描述
DeleteRecordingJobRequest	JobToken 判定该删除的录制任务。 tt:RecordingJobReference JobToken[1][1]
DeleteRecordingJobResponse	空消息。
故障码	描述
env:Sender ter:InvalidArgVal ter:NoRecordingJob	JobToken 不指定一个现有的任务。

19.15 获取录制任务集

GetRecordingJobs 将返回一个设备所有录制任务的清单。

表 271：GetRecordingJobs 命令

GetRecordingJobs	请求-响应
消息名称	描述
GetRecordingJobsRequest	空消息。
GetRecordingJobsResponse	JobItem 标识设备的任务，并持有任务的当前配置。 tt:GetRecordingJobsResponseItem JobItem [0][unbounded]
故障码	描述
没有具体的故障码。	

19.16 设置录制任务配置

SetRecordingJobConfiguration 将更改录制任务的配置。

表 272：SetRecordingJobConfiguration 命令

SetRecordingJobConfiguration	请求-响应
消息名称	描述
SetRecordingJobConfigurationRequest	从 SetRecordingJobConfiguration 返回的任务配置应和任务配置传递到 SetRecordingJobConfiguration 的相同，除了接收器令牌和自动创建接收器。在返回的结构中，接收器令牌应当有效且应忽略自动创建接收器。 tt:RecordingJobReference JobToken[1][1] tt:RecordingJobConfiguration JobConfiguration [1][1]
SetRecordingJobConfigurationResponse	JobConfiguration 结构是设备使用的设置。JobConfiguration 将与 CreateRecordingJob 有所不同。 tt:RecordingJobConfiguration JobConfiguration [1][1]
故障码	描述
env:Sender ter:InvalidArgVal ter:NoRecordingJob	JobToken 不指定现有的任务。
env:Sender ter:InvalidArgVal	JobConfiguration 内容无效。

ter:BadConfiguration	
env:Receiver ter:Action ter:MaxReceivers	AutoCreateReceivers 标志为真，如果接收服务不能建立一个新的接收，此错误将返回。

SetRecordingJobConfiguration 将删除任何自动建立的、不再用做改变录制任务设置结果的接收器对象。

19.17 获取录制任务配置

GetRecordingJobConfiguration 将返回录制任务的当前配置。

表 273：GetRecordingJobConfiguration 命令

GetRecordingJobConfiguration	请求-响应
消息名称	描述
GetRecordingJobConfigurationRequest	JobToken 判定将被检索设置的录制任务。 tt:RecordingJobReference JobToken[1][1]
GetRecordingJobConfigurationResponse	JobConfiguration 保持当前录制任务的设置。 tt:RecordingJobConfiguration JobConfiguration [1][1]
故障码	描述
env:Sender ter:InvalidArgVal ter:NoRecordingJob	JobToken 不指定一个现有的任务。

19.18 设置录制模式

SetRecordingJobMode 将改变录制任务的模式。使用这种方式相当于检索录制任务的设置，然后用不同的方式回写。

表 274：SetRecordingJobMode 命令

SetRecordingJobMode	请求-响应
消息名称	描述
SetRecordingJobModeRequest	JobToken 判定将被改变录制模式的录制任务。Mode 必须是录制任务的新模式。 tt:RecordingJobReference JobToken[1][1] tt:RecordingJobMode Mode[1][1]
SetRecordingJobModeResponse	空消息。
故障码	描述
env:Sender ter:InvalidArgVal ter:NoRecordingJob	任务令牌不指定一个现有的任务。
env:Sender ter:InvalidArgVal ter:BadMode	模式无效。

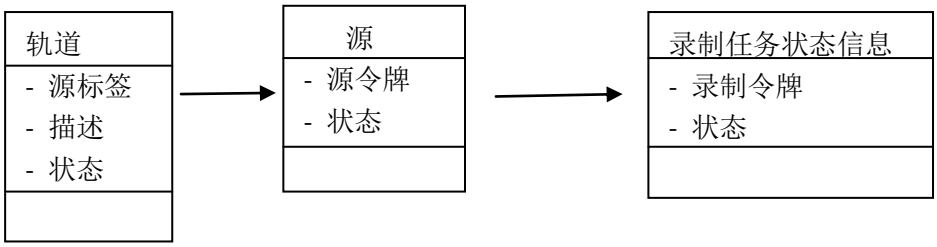
19.19 获取录制任务状态

GetRecordingJobState 返回录制任务的状态。包含聚合的状态以及每个录制任务轨道的状态。

表 275：GetRecordingJobState 命令

GetRecordingJobState	请求-响应
消息名称	描述
GetRecordingJobStateRequest	JobToken 判定哪一个录制任务将被获取状态。 tt:RecordingJobReference JobToken[1][1]
GetRecordingJobStateResponse	State 将保持录制任务的状态。 tt:RecordingJobStateInformation State[1][1]
故障码	描述
env:Sender ter:InvalidArgVal ter:NoRecordingJob	JobToken 不指定一个现有的任务。

录制任务的状态信息结构包含以下图例：



RecordingToken：录制任务记录的录制鉴定。

State：（如部分录制任务状态信息）将持有对整个录制任务信息结构的聚集状态。

SourceToken：判定录制任务的数据源。

State：（如部分录制任务状态源）将持有对整个录制任务状态源的聚集状态。

SourceTag：要判定数据源的轨道提供的数据。

Destination：判定目标轨道。

State：（如部分录制任务轨道状态）应提供轨道的工作状态。有效值状态应当是“Idle”，“Active”和“Error”。如果状态是“Error”，错误的地方可能入了一个已经定义的值。

Error：如果存在，将说明已经定义了的字符串值的错误。该字符串应是英文。

设备适用于以下规则计算聚合状态：

Idle	所有子节点的状态值是“闲置”
PartiallyActive	一些子节点的状态是“活动”，一些子节点是“闲置”
Active	所有子节点的状态值是“活动”
Error	至少有一个子节点有“错误”状态

19.20 事件

录制服务通过事件服务调度事件。它能产生这一章中列出的条件的事件时，触发事件发生。

有些类似自动生成的事件可以用 FindEvents 方法在搜索服务搜索。参见 20.17 节。

19.20.1 录制任务状态变化

如果录制任务状态信息结构的状态域改变，设备将发送以下事件：

```
        Topic: tns1:RecordingConfig/JobState
    <tt:MessageDescription IsProperty="true">
      <tt:Source>
    <tt:SimpleItemDescription Name="RecordingJobToken"
    Type="tt:RecordingJobReference"/>
      </tt:Source>
      <tt:Data>
        <tt:SimpleItemDescription Name="State" Type="xs:String"/>
   <tt:ElementItemDescription Name="Information"
    Type="tt:RecordingJobStateInformation"/>
  </tt:Data>
    </tt:MessageDescription>
```

19.20.2 设置变化

如果录制的设置变化，设备将发送以下事件：

```
        Topic: tns1:RecordingConfig/RecordingConfiguration
    <tt:MessageDescription IsProperty="false">
      <tt:Source>
        <tt:SimpleItemDescription Name="RecordingToken" Type="tt:RecordingReference"/>
      </tt:Source>
      <tt:Data>
       <tt:ElementItemDescription Name="Configuration" Type="tt:RecordingConfiguration"/>
  </tt:Data>
    </tt:MessageDescription>
```

如果轨道的设置变化了，设备将发送以下事件：

```
    Topic: tns1:RecordingConfig/TrackConfiguration
        <tt:MessageDescription IsProperty="false">
  <tt:Source>
            <tt:SimpleItemDescription                    Name="RecordingToken"
        Type="tt:RecordingReference"/>
```

```
            <tt:SimpleItemDescription Name="TrackToken" Type="tt:TrackReference"/>
          </tt:Source>
          <tt:Data>
            <tt:ElementItemDescription                                    Name="Configuration"
        Type="tt:TrackConfiguration"/>
  </tt:Data>
        </tt:MessageDescription>
```

如果录制任务的设置变化了，设备将发送以下事件：

```
      Topic: tns1:RecordingConfig/TrackConfiguration
      <tt:MessageDescription IsProperty="false">
        <tt:Source>
          <tt:SimpleItemDescription Name="RecordingToken" Type="tt:RecordingReference"/>
    <tt:SimpleItemDescription Name="TrackToken" Type="tt:TrackReference"/>
        </tt:Source>
        <tt:Data>
          <tt:ElementItemDescription Name="Configuration" Type="tt:TrackConfiguration"/>
  </tt:Data>
      </tt:MessageDescription>
      </tt:Data>
      </tt:MessageDescription>
```

19.20.3 删除数据

当数据删除时，设备将发送以下事件：

```
      Topic: tns1:RecordingConfig/DeleteTrackData
      <tt:MessageDescription IsProperty="false">
        <tt:Source>
          <tt:SimpleItemDescription Name="RecordingToken" Type="tt:RecordingReference"/>
    <tt:SimpleItemDescription Name="TrackToken" Type="tt:TrackReference"/>
        </tt:Source>
        <tt:Data>
          <tt:SimpleItemDescription Name="StartTime" Type="xsDateTime"/>
          <tt:SimpleItemDescription Name="EndTime" Type="xsDateTime"/>
  </tt:Data>
      </tt:MessageDescription>
```

19.20.4 录制和轨道的建立与删除

当录制建立时，设备将发送以下事件：

```
    Topic: tns1:RecordingConfig/CreateRecording
<tt:MessageDescription IsProperty="false">
    <tt:Source>
        <tt:SimpleItemDescription Name="RecordingToken" Type="tt:RecordingReference"/>
     </tt:Source>
```

```
    <tt:Data>
    </tt:Data>
  </tt:MessageDescription>
```

当录制删除时，设备将发送以下事件：

```
    Topic: tns1:RecordingConfig/DeleteRecording
  <tt:MessageDescription IsProperty="false">
  <tt:Source>
      <tt:SimpleItemDescription Name="RecordingToken" Type="tt:RecordingReference"/>
    </tt:Source>
    <tt:Data>
    </tt:Data>
  </tt:MessageDescription>
```

当轨道建立时，设备将发送以下事件：

```
    Topic: tns1:RecordingConfig/CreateTrack
  <tt:MessageDescription IsProperty="false">
  <tt:Source>
      <tt:SimpleItemDescription Name="RecordingToken" Type="tt:RecordingReference"/>
    <tt:SimpleItemDescription Name="TrackToken" Type="tt:TrackReference"/>
    </tt:Source>
    <tt:Data>
    </tt:Data>
  </tt:MessageDescription>
```

当轨道删除时，设备将发送以下事件：

```
    Topic: tns1:RecordingConfig/DeleteTrack
<tt:MessageDescription IsProperty="false">
    <tt:Source>
      <tt:SimpleItemDescription Name="RecordingToken" Type="tt:RecordingReference"/>
    <tt:SimpleItemDescription Name="TrackToken" Type="tt:TrackReference"/>
    </tt:Source>
    <tt:Data>
    </tt:Data>
  </tt:MessageDescription>
```

19.21 示例

19.21.1 例1：单摄像头的安装录制

有两个步骤，第一步是设置视频服务器

```
  ; Create recording (this implicitly creates an A, V and M track)
  RecordToken = CreateRecording(RecordConfiguration)

  ; The tracktokens are predefined. We don’t have to find them on the device
  TrackToken1 = “VIDEO001”
```

TrackToken2 = “AUDIO001”
TrackToken3 = “META001”

; Create a recording job, assume that we set mode to idle, auto create receiver
JobToken, ActualJobConfig = CreateRecordingJob(JobConfiguration)

; Configure the receiver
ConfigureReceiver(ActualJobConfiguration.ReceiverToken,ReceiverConfiguration)

这一步完成配置。

最后，调用一些实体开始录制

; Activate the recording job
SetRecordingJobMode(JobToken, Active)

让任务运行。这将导致视频服务器建立 RTSP 连接设备。

因此，启动和停止录制，只需要预先对录制任务呼叫 SetRecordingJobMode。由于嵌入式组态对象持续，配置周期只需要做一次。

19.21.2 例2：从一台摄像机录制多个流到一个单录制

这个例子非常类似的例子 1。该任务设置将保持两个接收器对象的参考。每个对象被配置为接收来自同一装置的不同的流。

; Create recording (this implicitly creates an A, V and M track)
RecordToken = CreateRecording(RecordConfiguration)

; The tracktokens are predefined. We don’t have to find them on the device
TrackToken1 = “VIDEO001”
TrackToken2 = “AUDIO001”
TrackToken3 = “META001”

; Create three additional tracks
TrackToken4 = CreateTrack(RecordToken, AudioConfig)
TrackToken5 = CreateTrack(RecordToken, VideoConfig)
TrackToken6 = CreateTrack(RecordToken, MetadataConfig)

; Create a recording job, assume that we set mode to idle,auto create two receivers
JobToken, ActualJobConfiguration = CreateRecordingJob(JobConfiguration)

; Configure the receivers
ConfigureReceiver(ActualJobConfiguration.ReceiverToken[1], Receiver1Configuration)
ConfigureReceiver(ActualJobConfiguration.ReceiverToken[2], Receiver2Configuration)

调用一些实体并真正开始录制

; Activate the recording job
SetRecordingJobMode(JobToken, Active)

20 记录搜索

20.1 介绍

搜索服务提供大量的操作，用于查找记录器内感兴趣的数据。搜索最常用的方式是搜索事件，事件就是包含元数据轨道的记录或在设备中的其他相关记录

读取记录摘要（GetRecordingSummary）返回一个包含所有记录的摘要，用于提供一个时间轴的浏览。

读取记录信息（GetRecordingInformation）返回一条记录的信息，如开始时间和当前状态。

读取媒体属性（GetMediaAttributes）返回在指定时间的一个记录的媒体属性。

实际的搜索是由查找和结果操作组成的，每一个找操作启动一个搜索对话，客户端就可以从查找对话中用增量的方式获取结果，或者同时，依赖实现和范围搜索。有四对查找操作分别对应记录，记录事件，PTZ 位置和元数据.

读取搜索状态（GetSearchState）返回一个搜索对话的状态。

结束搜索（EndSearch）结束一个搜索对话，停止搜索，返回和阻塞结果操作。

20.2 概念

20.2.1 搜索方向

搜索被执行是从时间线上的出发点到一个结束点，假如结束点在出发点之前，则会执行逆序搜索，这会非常的有用，假如仅对最新的匹配事件感兴趣，或者方便读取结果在最新到最老的命令。

假如没有指定结束点，查找将会从出发点一直执行下去。

20.2.2 记录事件

描述一个与记录相关的离散事件，把它说成是一个通知消息，但这并不一定意味着它已经作为一个通知被记录。

记录事件可以是一个包含记录元数据跟踪的通知，可以通过记录装置创建它，作为一个内部事件或机制的结果，或者客户端通过用网络服务请求或元数据流的方式，把记录事件嵌入通知。然而当记录事件已经被创建并且与特定的记录相关联时，这个规范没有规定数据在设备中的存储格式，它仅关心数据在接口上的表达格式。

然而，创建和记录事件总是被看做与查找滤波和结果反馈相关的通知。每一个记录事件有一个通知主题，详见 15.7 代码规范。预先定义的记录事件，详见 20.17 章节。

与属性事件的初始状态通信，虚拟的开始状态事件能够被返回在查找结果中，包含查找区间起始点的一个或多个属性值。这样的开始状态在场景中都是虚拟的事件，他们是在服务正在进行时创建的，而不是从记录数据收集的。假如客户端表明，希望给这样的事件设定适当的标记，那么定义在查找滤波器中匹配主题的虚拟事件，应该返回在查找范围内的一些记录。

20.2.3 查找对话

通过一个找操作开始一个异步的搜索对话，然后通过对话的搜索节点鉴定身份。读取结果操作是关于找操作创建的对话，使用它可以增量式的返回结果。查找可以在三种情况下终止：

KeepAlive time expires –假如从客户端没有请求发出，那么在指定的时间间隔内指定的对话就将会终止。

通过读取结果方式返回搜索对话最后的数据，显示搜索状态的结果为“完成”，搜索终止。

EndSearch –客户端明确的结束一个对话。

结束一个对话将取消正在进行的搜索，，立即返回和进一步请求这个终止了的对话，就会产生错误的消息。一个设备不应该重复使用某个搜索标识符，因为它会误导客户端，到底一个结束对话是否被终止。

20.2.4 查找范围

范围包含大量的操作元素，用于限制搜索的数据范围。

20.2.4.1 包括的数据

通过给每个主题指定一系列的标志符，客户端能够随意的定义源和记录搜索， 假如给定几个主题，那么指定的这些标识符就会被用到。假如没有指定源或记录的标识符，那么就代表所有记录。通过记录信息滤波器可以进一步改善范围，但是，当记录被指定，滤波器就只会运用于记录的子集。

20.2.4.2 记录信息滤波器

相对于指定一系列的记录标识符，通过操作记录信息结构的 XPath 滤波器，记录可以被滤波。通过使用 20.18 中定义的 XPath 方式来比较，客户可以对记录信息结构中的所有元素进行过滤。假如一个记录信息滤波被供应，只有匹配滤波器的记录可以成为范围的部分。例如 一个在查找范围内仅仅包含音频的记录滤波器：

boolean(//Tracks[TrackType = “Audio”])

20.2.5 搜索过滤器

搜索过滤器用于指定要搜索的类。分别见找事件，找 PTZ 位置，找元数据，它们的作用域为规定的记录范围。

20.3 数据结构

20.3.1 记录信息结构

记录信息包含记录的信息，组成的轨道和源：

RecordingToken –记录的唯一标识符

EarliestRecording –最早记录的数据和时间

LatestRecording –最新记录的数据和时间
Content –内容的描述信息
RecordingStatus –记录的当前状态可以随时开始，停止，移动和移除
RecordingSourceInformation –一个包含记录的源信息的结构
TrackInformation –一系列跟踪信息结构

20.3.2 记录源信息结构

记录源的信息包含：
SourceId –由创建记录的客户端选择的一个源标识符，这个标识符对 NVS 不透明，客户端可以对其使用任何类型的 URI。
Name –源信息名称
Location –源位置的信息描述
Description –源信息描述
Address –源的信息 URI

20.3.3 跟踪信息结构

在记录中一个单轨信息包含：
TrackToken –轨道的标识符，轨道标准符在一个记录中用到的所有轨道标识中是独一无二的
TrackType –轨道类型的标识符（视频，音频或元数据）
Description –轨道的信息描述
DataFrom –轨道中最老记录的数据和时间
DataTo –轨道中最新记录的数据和时间

20.3.4 列举查找状态

查找状态可以是下面的其中一个：
Queued –查找从没开始
Searching –查找正在进行中，能够产生新的结果
Completed –查找完成，将不会有新的结构产生

20.3.5 媒体属性结构

媒体属性包含媒体轨道信息，就是在一个特定时间范围的特定记录的信息。时间范围可以是一个时间点，即结束和起始点是重合的。
RecordingToken –结构关心的记录
From –一个有效地，记录在案的起始点
Until –一个有效地记录在案的终点
VideoAttributes –视频属性集，，描述一个记录的视频跟踪数据
AudioAttributes –音频属性集，，描述一个记录的音频跟踪数据
MetadataAttributes –元数据属性集，，描述一个记录的元数据跟踪数据

20.3.6 找事件结果结构

RecordingToken –标识找到事件的记录
TrackToken –标识找到事件的轨道
Time –找到事件的数据和时间
Event –要找的事件消息
StartStateEvent –假如为真，表示事件代表在记录中一个或多个属性的开始状态

20.3.7 找PTZ位置结果结构

RecordingToken –标识包含匹配位置的记录
TrackToken –标识包含匹配位置的轨道
Time –匹配位置的数据和时间
Position –匹配的 PTZ 矢量

20.3.8 PTZ位置过滤结构

包含必要的元素来说明在 PTZ 位置找什么，PTZ 矢量应该与存储在记录中 PTZ 坐标空间相一致
MinPosition – 查找的 PTZ 位置的下边界
MaxPosition –查找的 PTZ 位置的上边界
EnterOrExit –假如为真，当进入或退出指定的 PTZ 量时查找

20.3.9 元数据过滤结果

包含一个运用于元数据流结构的 XPath 符号。
一个查找重叠在右下象限的场景的对象符号的例子：
boolean(//Object/Appearance/Shape/BoundingBox[@right > ”0.5”])
and boolean(//Object/Appearance/Shape/BoundingBox[@bottom
> ”0.5”])

20.3.10 找元数据结果结构

RecordingToken –识别包含匹配的元数据记录
TrackToken –识别包含匹配元数据的轨道
Time –匹配元数据的数据和时间

20.4 获取记录概要（GetRecordingSummary）

GetRecordingSummary 是用于读取所有记录数据的概要描述。这个操作支持所有执行记录查找服务的设备。

表 276：读取记录信息概要命令

读取记录概要命令（GetRecordingSummary） 请求应答（Request-Response）	
消息名字	描述
读取记录概要请求 (GetRecordingSummaryRequest)	空消息
读取记录概要响应 （GetRecordingSummaryResponse）	返回一个结构体包含：当设备中有一个数据记录时指定的数据起始点时间 DataFrom，终点时间 DataUntil，估算总共的记录和跟踪数。 tt:RecordingSummary summary[1][1]
故障代码	描述
	没有命令存在

20.5 读取记录信息（GetRecordingInformation）

GetRecordingInformation 返回一条由记录标识符指定的记录信息。执行记录查找服务的设备应该支持这个操作。

表 277：读取记录信息命令

读取记录信息命令（GetRecordingInformation） 请求应答（Request-Response）	
消息名字	描述
读取记录信息请求 (GetRecordingInformationRequest)	请求描述 tt:ReferenceToken RecordingToken [1][1]
读取记录信息响应 （GetRecordingInformationResponse）	响应描述 tt:RecordingInformation RecordingInformation[1][1]
故障代码	描述
env:Sender ter: InvalidArgVal ter: InvalidToken	记录标识符无效

20.6 读取媒体属性（GetMediaAttributes）

GetMediaAttributes 返回一系列的媒体属性，包括所有特定时间内特定记录轨道的属性，用这个操作的客户端可以把它当做一个非阻塞的操作来使用。设备应该设置 MediaAttributes 结构体中的 starttime 和 endtime，用于估算是否这个范围将会引起操作阻塞，详见 MediaAttributes 结构了解更多信息。执行记录查找服务的设备都应该支持这个操作。

表 278：读取媒体属性

读取媒体属性命令（GetMediaAttributes） 请求应答（Request-Response）	
消息名字	描述
读取媒体属性请求 (GetMediaAttributesRequest)	RecordingTokens 是一系列需要查询的记录标识符，Time 是请求查询的信息时间范围 tt:ReferenceToken RecordingTokens [0][unbounded] xs:dateTime Time[1][1]

读取媒体属性响应 （GetMediaAttributesResponse）	包含一个 MediaAttributes 结构体对于每个请求的记录标识符 tt:MediaAttributes MediaAttributes [0][unbounded]
故障代码	描述
env:Sender ter:InvalidArgVal ter:InvalidToken	记录标识符无效

20.7 找记录（FindRecordings）

FindRecordings 开始一个搜索对话，寻找与请求匹配的记录（见 20.2.4），通过 GetRecordingSearchResults 命令可以获取搜索结果，GetRecordingSearchResults 请求指定请求返回数据的查找的标识符。

当设备发生以下情况时，停止查找：

整个时间范围从 StartPoint 到 EndPoint 都已经查找完毕

已经达到定义的 MaxMatches 参数

客户执行 EndSession 请求来停止对话

与对话相关的 KeepAliveTime 使对话终止

结果数量未被定义，那么就允许设备返回任何数量的结果，执行记录查找服务的设备都应该支持这个命令。

表 279：找记录命令

找记录命令（FindRecordings）	请求应答（Request-Response）
消息名字	描述
找记录请求 (FindRecordingsRequest)	SearchScopeScope 定义数据集的查找范围，当到达 MaxMatches 时，停止查找。KeepAliveTime 对话超时时间 tt:SearchScopeScope [1][1] xs:int MaxMatches [0][1] xs:duration KeepAliveTime [1][1]
找记录响应 （FindRecordingsResponse）	返回查找标识符识别请求的查找对话 tt:JobToken SearchToken [1][1]
故障代码	描述
env:Receiver ter:Action ter:ResourceProblem	设备不能创建新的查找对话

20.8 获取记录搜索结果（GetRecordingSearchResults）

GetRecordingSearchResults 命令用于获取执行 FindRecordings 命令得到的记录查找对话中的结果。响应应该不包含在相同对话框中请求返回过的结果。假如设定了最大结果，响应就不应该大于最大值。

GetRecordingSearchResults 应该被阻塞，除非：

假如 MaxResults 被设定，对于响应，MaxResults 的结果都是可用的

假如 MinResults 被设定，对于响应，MinResults 的结果都是可用的
WaitTime 期满
查找完成或停止
执行记录查找服务的设备都应该支持这个操作。

表 280：读取记录查找结果命令

找记录命令（GetRecordingSearchResults）	请求应答（Request-Response）
消息名字	描述
取记录查找结果请求 (GetRecordingSearchResultsRequest)	SearchToken 指定查找对话，MinResults 定义了返回的最小结果数，在完成查找对话后，如果结果数小于最小结果数，所有的结果应该被返回，MaxResults 定义了返回的最大结果数，WaitTime 定义了最长等待结果时间阻塞。 tt:JobToken SearchToken [1][1] xs:int MinResults [0][1] xs:int MaxResults [0][1] xs:duration WaitTime [0][1]
取记录查找结果响应 （GetRecordingSearchResultsResponse）	返回一个结构体，包含当前的查找状态和一系列 RecordingInformation 结构体 tt:FindRecordingResultList ResultList[1][1]
故障代码	描述
env:Sender ter:InvalidArgVal ter:InvalidToken	查找标识符无效

20.9 找事件（FindEvents）

FindEvents 开始一个找对话，寻找与请求匹配的记录事件（见 20.2.2），查找的结果通过 GetEventSearchResults 命令来获取，GetEventSearchResults 指定请求返回数据查找的标识符。
当设备发生以下情况时，停止查找：
整个时间范围从 StartPoint 到 EndPoint 都已经查找完毕
MaxMatches 参数定义的匹配总数已经被找到
客户执行 EndSession 请求来停止对话
与对话相关的 KeepAliveTime 期满使对话终止
结果应该按照时间排序，在向前查找中升序，在向后查找中降序，执行记录查找服务的设备应该支持这个操作。

表 281：找事件命令

找事件命令（FindEvents）	请求应答（Request-Response）
消息名字	描述
找事件请求 (FindEventsRequest)	StartPoint 是查找时间的起点，EndPoint 是查找结束的时间点，这个时间点可以在 StartPoint 之前，因为在某些情况下是反向查找的 假如 EndPoint 被忽略，那么它会向前查找到回到 StartPoint，Scope 定义了搜索的数据集，SearchFilter 包括主题和消息过滤，在查找时应定义查找

	对象，通过设定 IncludeStartState 为真，表示在起始点为符合查找要求的虚拟事件，那么客户端允许返回其在记录中的状态，当达到 MaxMatches 值时，查找结束，KeepAliveTime 是每个请求对话后的对话超时。 xs:dateTime StartPoint [1][1] xs:dateTime EndPoint [0][1] tt:SearchScopeScope [1][1] tt:EventFilter SearchFilter [1][1] xs:boolean IncludeStartState [1][1] xs:int MaxMatches [0][1] xs:duration KeepAliveTime [1][1]
找事件响应 （FindEventsResponse）	返回 SearchToken 识别请求创建的查找对话 tt:JobToken SearchToken [1][1]
故障代码	描述
env:Receiver ter:Action ter:ResourceProblem	设备不能创建一个新的查找对话

20.10 读取事件搜索结果（GetEventSearchResults）

GetEventSearchResults 命令用于获取从先前用 FindEvents 命令而执行的记录查找对话中的结果。响应应该不包括已经在先前在相同对话框中请求返回过的结果。假如设定了 MaxResults，响应就不应该大于 MaxResults。

GetEventSearchResults 应该被阻塞，直到：

假如 MaxResults 被设定, MaxResults 的结果都是可用的对于响应

假如 MinResults 被设定, MinResults 的结果都是可用的对于响应

WaitTime 期满

查找完成或停止

执行记录查找服务的设备都应该支持这个操作。

表 282：读取事件搜索结果

读取事件搜索命令（GetEventSearchResults） 请求应答（Request-Response）	
消息名字	描述
读取事件搜索结果请求 (GetEventSearchResultsRequest)	SearchToken 定义了查找对话，MinResults 定义了返回的最小结果数，在完成查找对话后，如果结果数小于 MinResults，所有的结果应该被返回，MaxResults 定义了返回的最大结果数，WaitTime 定义了阻塞的最长等待结果时间。 tt:JobToken SearchToken [1][1] xs:int MinResults [0][1] xs:int MaxResults [0][1] xs:duration WaitTime [0][1]
读取事件搜索结果响应	返回一个结构体，包含当前的 SearchState 和一系列

（GetEventSearchResultsResponse）	FindEventResult 的结构体。 tt:SearchState SearchState [1][1] tt:FindEventResult FindEventResult [0][unbounded]
故障代码	描述
env:Sender ter:InvalidArgVal ter:InvalidToken	查找标识符无效

20.11 查找 PTZ 位置（FindPTZPosition）

FindPTZPosition 开始于一个找对话，寻找匹配请求中定义的范围的 PTZ 位置（见 20.2.2），查找的结果通过读取 PTZ 位置查找结果请求来获取，GetPTZPositionSearchResults 请求指定返回数据查找的标识符。

当设备发生以下情况时，停止查找：

整个时间范围从 StartPoint 到 EndPoint 都已经查找完毕

MaxMatches 参数定义的匹配的总数已经被找到

客户执行 EndSession 请求来停止对话

与对话相关的 KeepAliveTime 期满使对话终止

在设备的任何记录中，元数据跟踪无论何时 CanContainPTZ 为真时，设备就应该支持这个操作。

表 283：查找 PTZ 位置命令

找 PTZ 位置命令（FindPTZPosition）	请求应答（Request-Response）
消息名字	描述
找 PTZ 位置请求 (FindPTZPositionRequest)	StartPoint 是查找时间的起点，EndPoint 是查找结束的时间点，这个时间点可以在开始点之前，因为在某些情况下是反向查找的 假如结束点被忽略，那么它会向前查找直到回到 StartPoint，Scope 定义了搜索的数据集，SearchFilter 包含了搜索条件，搜索条件定义了搜索的 PTZ 位置；通过设定包含开始状态为真，虚拟的事件在起始点若符合查找要求的应该被返回，当达到 MaxMatches 值时，搜索结束，KeepAliveTime 是每个请求对话后的对话超时。 xs:dateTime StartPoint [1][1] xs:dateTime EndPoint [0][1] tt:SearchScopeScope [1][1] tt:PTZPositionFilter SearchFilter [1][1] xs:int MaxMatches [0][1] xs:duration KeepAliveTime [1][1]
找 PTZ 位置响应 （FindPTZPositionResponse）	返回 SearchToken 识别请求创建的查找对话 tt:JobToken SearchToken [1][1]
故障代码	描述
env:Receiver ter:Action	设备不能创建一个新的查找对话

ter:ResourceProblem	
env:Receiver ter:ActionNotSupported ter:NotImplemented	这个方式不执行

20.12 读取 PTZ 位置搜索结果（GetPTZPositionSearchResults）

GetPTZPositionSearchResults 命令用于获取从先前 FindPTZPosition 命令而执行的 PTZ 位置查找对话中的结果。响应应该不包括已经在先前在相同对话框中请求返回过的结果。假如设定了 MaxResults，响应就不应该大于 MaxResults。

GetPTZPositionSearchResults 应该被阻塞，直到：

假如 MaxResults 被设定，对于响应， MaxResults 的结果都是可用的

假如 MinResults 被设定，对于响应，MinResults 的结果都是可用的

WaitTime 期满

查找完成或停止

在设备的任何记录中，元数据跟踪无论何时 CanContainPTZ 参数为真，设备都应该支持这个操作。

表 284：读取 PTZ 位置搜索结果

读取 PTZ 位置搜索搜索命令（GetPTZPositionSearchResults） 请求应答（Request-Response）	
消息名字	描述
读取 PTZ 位置搜索结果请求 (GetPTZPositionSearchResultsRequestt)	SearchToken 定义了查找对话，MinResults 定义了返回的最小结果数，在完成查找对话后，如果结果数小于 MinResults，所有的结果应该被返回，MaxResults 定义了返回的最大结果数，WaitTime 定义了最长等待结果时间阻塞。 tt:JobToken SearchToken [1][1] xs:int MinResults [0][1] xs:int MaxResults [0][1] xs:duration WaitTime [0][1]
读取 PTZ 位置搜索结果响应 （GetPTZPositionSearchResultsResponse）	返回一个结构体，包含当前的 SearchState 和一系列 FindPTZPositionResult tt:SearchState SearchState [1][1] tt:FindPTZPositionResult FindPTZPositionResult [0][unbounded]
故障代码	描述
env:Sender ter:InvalidArgVal ter:InvalidToken	查找标识符无效

20.13 查找元数据（FindMetadata）

FindMetadata 开始于一个找对话，在定义的范围内寻找匹配的元数据（见 20.2.2），查找的

结果通过 GetMetadataSearchResults 请求来获取，GetMetadataSearchResults 请求指定请求返回数据查找的标识符。
当设备发生以下情况时，停止查找：
整个时间范围从开始点到结束点都已经查找完毕
最大的匹配参数定义的匹配的总数已经被找到
客户执行终止对话请求来停止对话
与对话相关的最后请求期满使对话终止
假如 MetaDataSearch 功能设置为真，那么这个操作就会被强制支持，MetaDataSearch 功能是否为真，可以通过使用 GetCapabilities 命令来查看

表 285：查找元数据命令

找 PTZ 位置命令（FindMetadata）	请求应答（Request-Response）
消息名字	描述
找 PTZ 位置请求 (FindMetadataRequest)	StartPoint 是查找时间的起点，EndPoint 是查找结束的时间点，这个时间点可以在开始点之前，因为在某些情况下是反向查找的 假如结束点被忽略，那么它会向前查找到回到 StartPoint，Scope 定义了搜索的数据集，SearchFilter 包含了搜索条件，搜索条件定义了搜索的 PTZ 位置；通过设定包含开始状态为真，虚拟的事件在起始点若符合查找要求的应该被返回，当达到 MaxMatches 值时，搜索结束，KeepAliveTime 是每个请求对话后的对话超时。 xs:dateTime StartPoint [1][1] xs:dateTime EndPoint [0][1] tt:SearchScopeScope [1][1] tt:MetadataFilter SearchFilter [1][1] xs:int MaxMatches [0][1] xs:duration KeepAliveTime [1][1]
找 PTZ 位置响应 （FindMetadataResponse）	返回 SearchToken 识别请求创建的查找对话 tt:JobToken SearchToken [1][1]
故障代码	描述
env:Receiver ter:Action ter:ResourceProblem	设备不能创建一个新的查找对话

20.14 读取元数据搜索结果（GetMetadataSearchResults）

GetMetadataSearchResults 命令用于获取从先前用 FindMetadata 命令而执行的元数据查找对话中的结果。响应应该不包括已经在先前在相同对话框中请求返回过的结果。假如设定了 MaxResults，响应就不应该大于 MaxResults。
GetMetadataSearchResults 应该被阻塞，直到：
假如 MaxResults 被设定，对于响应，MaxResults 的结果都是可用的
假如 MinResults 被设定，对于响应，MinResults 的结果都是可用的

等待时间期满
查找完成或停止
假如 MetaDataSearch 功能设置为真，那么这个操作就会被支持，MetaDataSearch 功能是否为真在 SearchCapabilities 结构体中，可以通过使用 GetCapabilities 命令来查看

表 286：读取元数据搜索结果

读取元数据搜索结果命令（GetMetadataSearchResults） 请求应答（Request-Response）	
消息名字	描述
读取元数据搜索结果请求 (GetMetadataSearchResultsRequest)	SearchToken 定义了查找对话，MinResults 定义了返回的最小结果数，在完成查找对话后，如果结果数小于 MinResults，所有的结果应该被返回，MaxResults 定义了返回的最大结果数，WaitTime 定义了阻塞的最长等待结果时间。 tt:JobToken SearchToken [1][1] xs:int MinResults [0][1] xs:int MaxResults [0][1] xs:duration WaitTime [0][1]
读取元数据搜索结果响应 （GetMetadataSearchResultsResponse）	返回的结构体包含 SearchState 和一系列 FindMetadataResult 结构体 tt:SearchState SearchState [1][1] tt:FindMetadataResult FindMetadataResult [0][unbounded]
故障代码	描述
env:Sender ter:InvalidArgVal ter:InvalidToken	查找标识符无效

20.15 获取搜索状态（GetSearchState）

GetSearchState 返回指定搜索对话当前的状态：排队（Queued），搜索（Searching）或完成（Completed.）。对于执行记录搜索服务的设备，这个操作是强制的。

表 287：读取搜索状态命令

读取搜索状态命令（GetSearchState） 请求应答（Request-Response）	
消息名字	描述
读取搜索状态请求 (GetSearchStateRequest)	SearchToken 指定搜索对话 tt:JobToken SearchToken [1][1]
读取搜索状态响应 （GetSearchStateResponse）	返回的搜索对话当前的 State tt:SearchState State [1][1]
故障代码	描述
env:Sender ter:InvalidArgVal ter:InvalidToken	搜索标识符无效

20.16 结束搜索（EndSearch）

EndSearch 停止正在进行的搜索对话，请求返回一个阻塞结果，SearchToken 变成无效的。假如搜索在完成之前被打断，那么应该返回搜索到达的那个被打断时间点，假如搜索还没开始，那么应该返回开始点，假如搜索已经被完成了，那么由查找操作提供的终止点应该被返回，对于执行记录搜索服务的设备，这个操作是必须的。

表 288：结束搜索命令

结束搜索命令（EndSearch）	请求应答（Request-Response）
消息名字	描述
结束搜索请求 (EndSearchRequest)	SearchToken 指定搜索对话 tt:JobToken SearchToken [1][1]
结束搜索响应 （EndSearchResponse）	返回搜索结束时那个时间点 EndPoint xs:dateTime EndPoint [1][1]
故障代码	描述
env:Sender ter:InvalidArgVal ter:InvalidToken	搜索标识符无效

20.17 记录事件说明

于事件消息描述相对应，一个设备应该产生以下的事件。一个支持记录查找服务的设备应该记录这些消息通知，以便客户可以使用 FindEvents 命令来搜索这些消息。所有的记录事件都由设备生成，并嵌入历史记录中，而且历史记录应该有一个根目录 tns1:RecordingHistory

目录：tns1:RecordingHistory/Recording/State

```
<tt:MessageDescription IsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="RecordingToken"
Type="tt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:SimpleItemDescription Name="IsRecording" Type="tt:boolean"/>
</tt:Data>
</tt:MessageDescription>
```

当客户开始或停止记录指定的记录时，设备都会发送消息。在记录的开始，IsRecording 应该被设置为真。在停止记录时，IsRecording 应该设置为假。

目录：tns1:RecordingHistory/Track/State

```
<tt:MessageDescription IsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="RecordingToken"
Type="tt:ReferenceToken"/>
<tt:SimpleItemDescription Name="Track" Type="tt:ReferenceToken"/>
</tt:Source>
<tt:Data>
```

<tt:SimpleItemDescription Name="IsDataPresent" Type="tt:boolean"/>
</tt:Data>
</tt:MessageDescription>
当轨道数据出现时，也要发送信号，当数据出现时，就会发送一个 IsDataPresent 设置为真的消息，当数据无效时，就会发送一个 IsDataPresent 为假的消息，
一个 NVS 可以产生以下的事件，假如 NVS 支持这些事件，那么它就会自动的记录这些通知消息，以便客户端能够使用 FindEvent 命令来得到这些消息。
Topic: tns1:RecordingHistory/Track/VideoParameters
<tt:MessageDescription IsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="Recording" Type="tt:ReferenceToken"/>
<tt:SimpleItemDescription Name="Track" Type="tt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:SimpleItemDescription Name="VideoEncoding"
Type="tt:VideoEncoding"/>
<tt:SimpleItemDescription Name="VideoWidth" Type="xs:int"/>
<tt:SimpleItemDescription Name="VideoHeight" Type="xs:int"/>
<tt:SimpleItemDescription Name="tt:RateControl"
Type="VideoRateControl"/>
</tt:Data>
</tt:MessageDescription>

Topic: tns1:RecordingHistory/Track/AudioParameters

<tt:Source>
<tt:SimpleItemDescription Name="Recording" Type="tt:ReferenceToken"/>
<tt:SimpleItemDescription Name="Track" Type="tt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:SimpleItemDescription Name="AudioEncoding"
Type="tt:AudioEncoding"/>
<tt:SimpleItemDescription Name="AudioSampleRate" Type="xs:int"/>
<tt:SimpleItemDescription Name="AudioBitrate" Type="xs:int"/>
</tt:Data>
</tt:MessageDescription>
The NVS shall send either message (depending on the track data type) whenever any of these properties change.
当这些属性改变时，NVS 都应该送对应的消息（依赖于轨道数据类型）。

20.18 XPath 习惯用法

这个章节定义了 XPATH 语言，一个实现搜索服务的设备应该负责解析 XPath 字符串，XPath 字符串通过搜索服务的方法传送。

Dialect=http://www.onvif.org/ver10/tse/searchFilter

[1] Expression ::= BoolExpr | Expression 'and' Expression
| Expression 'or' Expression | '('Expression')' |
'not''('Expression')'
[2] BoolExpr ::= 'boolean''('PathExpr')' | 'contains''(' ElementPath ','
'"' String '"' ')'
[3] PathExpr ::= '//SimpleItem' NodeTest | '//ElementItem' NodeTest |
ElementTest
[4] NodeTest ::= '['AttrExpr']'
[5] AttrExpr ::= NameComp | ValueComp | AttrExpr 'and' AttrExpr | AttrExpr
'or' AttrExpr | 'not''(AttrExpr')'
[6] NameComp ::= NameAttr'='"'String'"'
[7] ValueComp ::= ValueAttr Operator '"'String'"'
[8] Operator ::= '=' | '!=' | '<' | '<=' | '>' | '>='
[9] NameAttr ::= '@Name'
[10] ValueAttr ::= '@Value'
[11] ElementTest ::= '/' ElementPath '['NodeComp']'
[12] ElementPath ::= ElementName ElementName*
[13] ElementName ::= '/' String
[14] NodeComp ::= NodeName Operator '"' String '"'
[15] NodeName ::= '@' String | String

例如一个 XPath 符号常常用于从基层查找记录，基层就是包含至少一条视频轨道的地方。

boolean(//Source[Location = “Basement”]) and
boolean(//Tracks[TrackType = “Video”])

21 重放控制

本节描述了 RTSP 重放记录流的使用，并且定义了一种服务（对 URI 使用 RTP 测绘其重放终端）

21.1 使用 RTSP 协议

重放控制是基于 RTSP [RFC 2326].的，然而因为它不支持一些由 CCTV 应用要求的功能，这个标准定义了一些扩展协议，下面是详细描述：

本标准在使用 RTSP 协议时做如下的规定：

1：服务器应该支持 RTP/RTSP/HTTP/TCP。这是一种媒体服务将支持用来实现媒体流的设备相同的传输协议，并且相同的要求将适用在重放流中。

2：服务器应该支持单播 RTP/UDP 传输流。

3：为了实现可靠的媒体数据包，客户应该使用基于 TCP 的传输重放。

4：在重播期间，服务器可能会不发送 RTSP 数据包。在典型的 RTSP 使用时，不要求数据包，因为通常采用的是一种可靠的传输的模式，并且因为绝对时间信息在流之内发送，使 RTSP 时间信息的发送者报告冗余信息。

21.1.1 RTSP描述

RTSP 返回的 SDP 描述命令应该包括跟踪参考，每一个跟踪记录允许客户端地图的曲目介绍在 SDP 记录的跟踪中。应该使用以下格式的标记：

a:x-onvif-track:<TrackReference>

For example:

NVS – NVT: DESCRIBE rtsp://192.168.0.1 RTSP/1.0

CSeq: 1

User-Agent: ONVIF Rtsp client

Accept: application/sdp

NVT – NVS: RTSP/1.0 200 OK

CSeq: 1

Content-Type: application/sdp

Content-Length: xxx

v=0

o= 2890842807 IN IP4 192.168.0.1

m=video 0 RTP/AVP 26

a=control:rtsp://192.168.0.1/video

a=x-onvif-track:VIDEO001

m=audio 0 RTP/AVP 98

a=control:rtsp://192.168.0.1/audio

a=x-onvif-track:AUDIO001

21.2 RTP 协议头部扩展

为了允许客户报告一个稳定和精确的时间戳（每个帧无论从哪个方向播放），有必要将确切的时间戳与数据包联系起来。或者每组数据包具有相同的时间戳。这是通过使用一个包含

NTP 时间戳的 RTP 头扩展来实现的，并且一些也需要实现重放的其他信息
重放机使用扩展标识 0xabac 来进行重播扩展
下面显示了一般形式的的 RTP 数据包包含的这些扩展：
表 289：RTP 包布局

<table>
<tr><td>V=2</td><td>P</td><td>X=1</td><td>CC</td><td>M</td><td>PT</td><td colspan="2">sequence number</td></tr>
<tr><td colspan="8">timestamp</td></tr>
<tr><td colspan="8">synchronization source (SSRC) identifier</td></tr>
<tr><td colspan="6">0xABAC</td><td colspan="2">length=3</td></tr>
<tr><td colspan="8">NTP timestamp…</td></tr>
<tr><td colspan="8">...NTP timestamp</td></tr>
<tr><td>C</td><td>E</td><td>D</td><td colspan="2">mbz</td><td colspan="2">CSeq</td><td>padding</td></tr>
<tr><td colspan="8">payload…</td></tr>
</table>

这个扩展领域如下：
. NTP 时间戳。一个 NTP[RFC1305]时间戳显示了 URC 与接入单元联系起来的绝对时间。
. C :1 位。表示该存取单元是一个同步点或干净点。比如，视频流的帧内编码的起始位。
. E :1 位。表示连续的一段记录的结尾。在每个录音的间隔，每个轨道的最后访问单元或者可得到的现有画面的结尾，都应该有这位的设置。当反向重放时，E 标志应该设置最后一帧在这段记录的结尾时。
.D ：1 位。表示该存取单元遵循如下的不连续传输。它主要在反向重放时被使用。每个 GOP 的包有个 D 位设置，因为它不按照以前的数据流顺序。（看 21.5 节）
.mbz：这一领域保留供以后使用但是必须为 0.
.CSep：1 个字节。这是低字节它使用的价值是用来启动传输。当客户发送多个连续播放命令。这个值可以用来决定每个新的 PLAY(播放)命令开始时数据从哪里来。
重放头的扩展应该在每个访问的第一个包中展现出来。它可能不会出现在随后的数据包接入单元。

21.2.1 NTP时间戳

该 RTP 扩展头的 NTP 时间戳应该单调递增，通过一个单一的实时传输协议流的数据包。如果有必要调整单调性的话。它们应该对应相应的钟表就像在原始发射机流中测量的那样兼容压缩头部扩展

21.2.2 压缩JEPG头扩展的兼容

重放头扩展可以与由压缩实时传输协议中使用的头扩展共存。有必要允许用此扩展，以便重放图像流。这个 JEPG 扩展是简单的附加到重播扩展中去的。它的存在表明一个实时传输协议报头的扩展长度字段大于 3，并且通过这个扩展，可以在开始的第四个字的启动码 0xFFD8 和 0xFFFF 中扩展内容。
以下说明了一些使用扩展的压缩包：
表 290：带有 JEPG 头布局的压缩包

V=2	P	X=1	CC	M	PT	sequence number
timestamp						
synchronization source (SSRC) identifier						
0xABAC						length=N+3
NTP timestamp…						
...NTP timestamp						
C	E	D	mbz	CSeq		padding
0xFFD8						jpeglength=N
extension payload: sequence of additional JPEG marker segments padded with 0xFF to the total extension length						
payload…						

21.3 RTSP 特性标签

重放服务使用了“ONVIF –REPLAY”的特征标签来显示它支持这个 RTP 协议的扩展，就像在这个标准里面描述的那样。这个允许客户为这些扩展查询服务器的支持通过使用请求头域，就像在 12.3.1 节描述的那样。

例子：

C->S: SETUP rtsp://server.com/foo/bar/baz.rm RTSP/1.0

CSeq: 302

Require: onvif-replay

S->C: RTSP/1.0 551 Option not supported

CSeq: 302

Unsupported: onvif-replay

这个重播服务器应该接受一个设置命令（包括一个包含“OIVIF-REPLAY”特征请求标签的请求头）

21.4 启动播放

回放通过数据播放的方法来启动。比如：

PLAY rtsp://192.168.0.1/path/to/recording RTSP/1.0

CSeq: 123

Session: 12345678

Require: onvif-replay

Range: clock=20090615T114900.440ZRate-

Control: no

ONVIF 设备可能支持逆向播放。逆向播放通过使用一个带有负价值的规模头领域来启动。例如为了没有数据丢失的完全重放，一个值-1.0 被使用。

PLAY rtsp://192.168.0.1/path/to/recording RTSP/1.0

CSeq: 123

Session: 12345678

Require: onvif-replay

Range: clock=20090615T114900.440ZRate-

Control: no

Scale: -1.0

如果一个设备支持逆向回放，那么它可以接受一个带有头值 1.0A 的规模头。除非这个速度控制头设置成了“no”，这个规模参数将在[RFC 2326]中描述的方式中使用。如果速度控制头设置成了“no”这个规模参数，一旦被显现出来，应该要么是 1.0 要是是—1.0，分别表示正向和反向播放。

21.4.1 领域范围

领域范围应该只能用绝对时间来表示。[RFC 2326]定义的其他格式不会被 ONVIF 重播的客户端使用。

服务器也可以选择支持其他的格式。通过使用[RFC 2326].中的 utc-range 绝对时间就被表示了出来。

无论打开关闭，范围都可以被用。在关闭的范围情况下，当向前播放和减少反播放，范围是要增加的（结束时间晚于开始时间）。变化的方向应该与这个规模头的值有关。在所有的情况下，变化的第一个点暗示出了回放的开始点。

例如：

PLAY rtsp://192.168.0.1/path/to/recording RTSP/1.0

CSeq: 123

Session: 12345678

Require: onvif-replay

Range: clock=20090615T114900.440Z-20090615T115000

Rate-Control: no

PLAY rtsp://192.168.0.1/path/to/recording RTSP/1.0

CSeq: 123

Session: 12345678

Require: onvif-replay

Range: clock=20090615T115000.440Z-20090615T114900

Rate-Control: no

Scale: -1.0

21.4.2 速度控制头领域

这个标准介绍了一个要么是“yes”要么是“no”的速度控制头领域。如果没有明确的显示这个域，那么就被假设成了“yes”，并且使用 RTP 定时机制，流是实时传输的。如果域是“no”，这个流将被尽快的传输，通过使用限制传输率的传输提供的流动控制。

在这两个码之间重要的不同是：“Rate-Control=yes”，由服务器来控制播放速率，而“Rate-Control=no”由客户端控制播放速率。速度控制回放将典型性的被 non-ONVIF 指定的客户使用，因为它们没有被指定为“Rate-Control=no”.当重放单记录的多轨道的时候，一个简单的 RTP 播放命令就启动了并且对速率不实施控制，来自轨道的数据应该被及时的混合，就像他们以相同的顺序被记录的时候。

21.4.3 帧头字段

帧头字段将用来减少正在传输的帧的数量，例如低带宽和处理负载，有三种可能码：

1：只有内部帧。这表明使用“intra”的价值选项必须遵照最小的间隔连续帧流。后者可以用来限制收到的帧的数量，即使存在一些接收器频繁请求 I-frames 而产生的“I-frame storms”。

2：只有内部帧和预测帧。这是通过使用“predicted”值表明的。这个值可以用来消除 B-frames，如果这个流包括它们。
3：所有帧。这是错误的。
例子：
为了只请求 intra 帧，
Frames: intra

为了请求 intra 帧以最小的 4000 毫秒为一个间隔
Frames: intra/4000

只请求 intra 帧和 predicted 帧
Frames: predicted

请求所有帧（不是必须明确指定此模式，而是考虑到被包括的这个例子的完整性）：
Frames: all
带有“intra”选项的区间参数被用来指代记录时间，不是播放时间；因此对于播放任何给定间隔相同的帧时，没有管播放速度。区间参数不应该显示出来，除非该帧的选项是“intra”服务器应该支持帧头域，这并不排除规模帧头域作为另外一种限制数据速度的方式的使用。这个规模帧头域的使用可能会在不同的服务器实现方式之间改变，正如[RFC 2326].中的那样。

21.4.4 同步点

21.5 回放

逆向重播将被启动，通过使用一个带有负值的规模头域就像上面描述的那样。

21.5.1 数据包传输顺序

逆向播放期间视频包传输的顺序是基于 GOPS 的

在逆向播放的期间，GOPS 将按逆向顺序被发送，但是一个 GOPS 里面的包的发送顺序却按前进方向。在它的 RTP 扩展头中，每个 GOP 的第一个包将有中断位设置。在它的 RTP 扩展头中，跟随一个间隔的 GOP 的最后一个包将有个 E 位设置。
当只传输关键的帧的时候，或者当编码器不是 motion-based 的（比如 JPEG）的时候，一个 GOP 被认为组成了一个单一的帧，但是仍然由多个包组成。数据包在每个帧之内，此情况下的数据包又按前进的顺序发送出去了。然而这些帧自己却按照反方向被发送。
音频和数据流将用这个视频流的命令镜像来传输，因此来自流的数据包将按相反的播放顺序方向发送，直到出现一个数据包的 D 位设置，在扩展头中(在不连续之前，它们跳到的那个点)。

21.5.2 RTP传输顺序号

在逆向传播期间，传输顺序中传输包的 RTP 序列号应该单调递增，不在预订的播放顺序中。

21.5.3 RTP时间戳

使用 RTP 时间戳是取决速度控制头的值，如果这个头的值是“no”（比如，客户控制播放速度），时间戳将从记录过的帧的采样时间中导出。如果速度控制头没有被显示或者是个“yes”值，这个 RTP 时间戳与播放时间相关，就像描述在[RFC 2326]中的那样。

如果速度控制是“no”，在逆向播放期间，包传输的的 RTP 时间戳应该与它们即将成为样子的相同（如果相同的包向前进方向传输的话）。不像那些序列号，RTP 时间戳与原始的记录顺序相关，而不是传输顺序。当流被记录的时候，服务器就接收到了的时间戳时，它们将用相同的时间戳。

这意味着单个的 GOP 的连续的 RTP 数据包将总是有个递增的时间戳。但是在逆向的播放期间，连续收到 GOPS 的指示框架的时间戳将减少。

在逆向播放期间，如果速度控制是“yes”，被传输的包的 RTP 时间戳应该显示每个框架并且应该呈现给客户端次数。因此单个的 GOP 的连续的包将减少这个 RTP 时间戳，并且在指示框架上的时间戳将增加。在这种模式下，依赖于速度的值的连续时间戳与规模头的间隔时间描述在[RFC 2326]附录 B。

21.6 RTSP 长连接

当控制被禁用和 RTP 流被置于底部，通过 RTSP 的连接(using the RTP/RTSP/TCP or RTP/RTSP/HTTP/TCP 传输)，客户端不会发送设置参数请求并且服务器不得超时这个连接，在没有这些要求的时候。这可能是因为客户端能接收这些要求的应答，例如播放被暂停了。从另一方面讲，为了确定其他的终端是否成为应答，在连接中，要么服务器要么客户端会使 TCP 网络保持活动。

21.7 当前记录片段

在当前的世界时间或者不久前，如果客户端开始播放，它可以实时的终止播放录像就像它被记录的那样。这种情况下服务器单纯的继续给客户端发送数据流就像它接收的那样。需要注意的是：E 位没有设置当前正在被记录的接入单元，虽然每一个访问单元发送到的重放客户端通常会被最后一个服务器识别的。如果记录停止，E 位将被设置成记录的最后一位访问单元。

21.8 结束片段

如果播放达到一个点之后（什么样的点：在这个流中不会再有数据被发送的点），它会停止发送数据，但不会进入暂停状态。发生这样的事情后，如果服务器恢复了记录，将用它接收到的新数据来重新恢复发送。

21.9 拖放

正如 10.5 章节的那样[RFC 2326]，当一个播放进程正在运行的时候，收到一个播放命令，它将不起作用直到当前的播放操作完成之后。这个规范增加了一个新的 RTSP 头（Immediate），“Immediate”用重播命令重写此行为靠得是使用：

PLAY rtsp://192.168.0.1/path/to/recording RTSP/1.0
CSeq: 123
Session: 12345678
Require: onvif-replay
Range: clock=20090615T114900.440ZRate-
Control: no
Immediate: yes

如果服务器收到了一个立即头文件设置为“yes”的播放命令，将立即从一个新的地方开始播放，取消任何存在的播放命令，从新的地方发送的第一个数据包，将有个 D 位设置在它的 RTP 扩展头中。

21.10 使用 RTCP 协议

服务器没有被要求发送 RTCP 包，如果它确实发送了，应该遵循如下的规则：

如果控制启用，实时传输控制的协议数据包应该被建立和传输，就像[RFC 3550]规范描述的那样。特别是，发送者的时间戳报告显示当前的时间挂表，并且不涉及的时间戳被嵌入在数据流的扩展标题中。

如果速度控制未被启用，每一个发送报告中的 NTP 和 RTP 时间戳应该被设置为零。

21.11 重放命令

本节描述重放服务提供的网络服务命令。

21.11.1 重放命令

GetReplayUri 请求一个 URI 可以用来启动一个播放记录流，通过 RTEP 的控制协议。URI 是有效的，仅仅在它指定的响应中。所有的重播服务实现方式应该支持 GetReplayUri 命令。

表 291：GetReplayUri 命令

GetReplayUri	请求-应答
消息名字	描述
GetReplayUriRequest	流设置包含了两部分，流类型定义是否单播或者组播媒体流被要求了，传输指定了一个传输协议链定义的媒体流通道通过不同的网络协议，这个 RecordingToken 暗示了这个记录被信息流化（视频流或者音频流）tt:StreamSetup StreamSetup [1][1] tt:ReferenceToken RecordingToken [1][1]
GetReplayUriResponse	包含利用 URI 来请求媒体流 xs:anyURI Uri [1][1]
误码	描述
env:Sender ter:InvalidArgVal ter:NoProfile	记录不存在
env:Sender ter:InvalidArgVal ter:InvalidStreamSetup	流类型的规范或者传输部分的流设置是不被支持的

env:Sender ter:OperationProhibited ter:StreamConflict	流类型的规范传输部分的流设置与其他流发生冲突

21.11.2 重播配置

重播配置结构包含重播服务的配置。在重播配置结构域是：

会话超时：

一个 RTP 会话具有保持时间。应该定期更新，以防止会话超时。如果会话超时，它应该被拆除。重播的会话超时遵循相同的规则就像应用现场流（通过使用媒体服务）在“实时流”章节讨论的那样。

21.11.3 设置重播配置

设置重播配置可以改变重播服务的配置。重播服务应该允许用这些命令来更改他的配置：

表 292：设置重播配置命令

设置重播配置	请求—应答
消息名字	描述
设置重播配置请求	配置应该为重播服务把握新配置 tt:ReplayConfiguration Configuration[1][1]
设置重播配置应答	空
误码	描述
env:Sender ter:InvalidArgVal ter:ConfigModify	该在配置中部能被设置

21.11.4 获取重播配置

获得重播配置返回当前重播服务的配置，重播应允许其配置检索，使用此命令。

表 293：获得重播配置命令

获得重播配置	请求-应答
消息名字	描述
获得重播配置请求	这应该是个空消息
获得重播配置应答	配置应该保存当前服务的配置 tt:ReplayConfiguration Configuration[1][1]
误码	描述

	没有命令指定误码

21.11.5 服务指定的误码

表 90 列出了重播服务指定的误码，此外，每个命令都能够产生一个通用的故障，看表 5。通用故障的字码作为一个具体的故障码被定义。双亲通用码在每一行的顶端下面是字码并且具体的故障字码在这个单元的底部。

表 294：具体的重播服务故障码

误码	Parent Subcode Subcode	错误的原因	描述
env:Sender	ter:InvalidArgVal ter:NoProfile	简介令牌不存在	被请求的简介令牌不存在
env:Sender	Sender ter:InvalidArgVal ter:InvalidStreamSetup	无效的流设置	流类型的规范和流设置的传输部分不被支持
env:Sender	ter:OperationProhibited ter:StreamConflict	流冲突	流类型的规范和流设置的传输部分与其他流引起冲突
env:Sender	ter:InvalidArgVal ter:ConfigModify	参数不能被设置	配置参数不能被设置

22 安全

就像对于所有的网络技术很实际的那样，网络视频通信的安全是最重要的考虑对象。网络安全的威胁是取决于实际的应用。然而某些应用程序最容易遭到攻击，其他的应用程序对它一点也不敏感。实施安全对策的成本变化取决于不同的攻击防范。这些事实意味着对于网络视频产品或者系统，我们不能列出一般的安全要求，但是可以尝试找到一个合理的安全要求的水平针对那些符合本规范的设备，并且试着定义最基本的允许安全网络视频系统的安全机制。

目前的规范从两个交流层面定义了安全机制

.传输层面的安全

.信息层面的要求

此规范采用的基本端口认证机制如下。

. IEEE 802.1X

22.1 传输层安全

传输层安全防止在客户和服务器之间数据发生改变。

传输层安全(TLS)被认为是一个比较成熟的标准，此标准通过加密为传输连接提供了一个基本水平的通信安全。

TLS 协议允许相互认证传输会话以及维护保密性和完整性的配置

一个符合本规范的设备应该支持 TLS 1.0[RFC2246]和相关的规范。设备应该支持 TLS 1.1[RFC4346].设备应该支持 TLS 1.2[RFC5426]。为了保护所有的 ONVIF 设备，设备还应该支持 TLS。为保护媒体流的 RTP/RTSP/HTTPS 的隧道选项，设备也要支持 TLS。这个规范概述了一个特定的 TLS 和其他相关的可以与 TLS 一起使用的规范的实现。

客户端也应该支持 TLS 1.0 [RFC 2246] 和 TLS 1.1 [RFC 4346].客户可能支持 TLS 1.2 [RFC 5246].

消息安全

22.1.1 支持密码套

支持 TLS 的设备也应该支持下面的所有的密码套 [RFC 2246], [RFC 3268]:

TLS_RSA_WITH_AES_128_CBC_SHA

TLS_RSA_WITH_NULL_SHA

如果一个客户支持 TLS，那么应该支持下面的密码套

TLS_RSA_WITH_AES_128_CBC_SHA

TLS_RSA_WITH_NULL_SHA

22.1.2 服务器身份验证

通过使用 TLS，支持 TLS 的设备应该支持服务器身份验证。设备应该支持认证 X.509 服务证书。RSA 密码的长度至少为 1024 位。客户端应该支持服务器身份验证，通过使用 TLS 本规范没有提供一个通用的完整的服务器证书和认证机构（CA）的模型。但是，设备管理命令可以进行证书检索和下载，定义在章节 8.4。

服务器的专用密码钥匙或者密码钥匙的安全引导机制的详情在本规范的范围之外。但是通用的键盘命令定义在章节 8.4。

22.1.3 客户端认证

支持 TLS 的设备应该支持用户认证协议。通过一个设备管理命令可以启用客户认证。定义在章节 8.4.

支持 TLS 的设备应该包含 RSA 认证类型并且应该支持验证 RSA 认证和客户签名。

客户端应该支持客户认证。如果支持客户认证，客户端将支持 RS 客户认证和签名和可以使用一个至少 1024 位的密码钥匙。

一个值得信赖的 CA 引导机制在目前的规范之外。此规范的未来版本可能定义标准化的引导机制。

22.2 消息安全

TLS 允许点对点的保密性和完整性。然而，网络服务要求一个更灵活的通信模式和中间节点，在这些解决办法中，TLS 还不能提供一个端到端的安全机制。此外，对于网络服务，

为了实现基于用户的访问控制级命令，有必要验证每个 SOAP 消息的起源。这个可以通过 WS 安全框架来完成。ONVIF WS 安全在章节 5.12 节中有简介。

22.3 IEEE802.1X

IEEE 802.1X 是一个 IEEE 标准，基于端口的网络接入控制的目的是提供身份认证和使设备连接到局域网端口的认证。它利用 IEEE802 的局域网的基础设施的物理访问特性为了提供一种验证和授权的设备连接到局域网端口的方法，此方法有点对点的连接特性，并且禁止访问此端口，万一认证和授权失败的时候。

这个规范推荐使用 IEEE 802.1X 来进行基于端口的无线网络认证。作为一个支持 EAP 的方法，支持 IEEE802.1X 的设备应该支持 EAP-PEAP/MSCHAPv2 类型。此设备也可能支持其他的 EAP 方法诸如 EAP-MD5, EAP-TLS and EAP-TTLS 类型

这个规范定义了一套命令来配置和管理 IEEE 802.1X 的配置，请参阅章节 8.4.7。

附件 A

附件 A.1 媒体配置主题

下列的媒体配置的实例，ONVIF 主题名字空间提供了一下的主题：

tns1:MediaConfiguration/Profile
tns1:MediaConfiguration/VideoSourceConfiguration
tns1:MediaConfiguration/AudioSourceConfiguration
tns1:MediaConfiguration/VideoEncoderConfiguration
tns1:MediaConfiguration/AudioEncoderConfiguration
tns1:MediaConfiguration/VideoAnalyticsConfiguration
tns1:MediaConfiguration/PTZConfiguration
tns1:MediaConfiguration/MetaDataConfiguration

每一个这些主题都代表了一个财产，客户端每订阅一个这些议题将被通知改变，创建和删除相应的实体。

不同主题的消息结构通过使用消息描述语言（介绍在章节 15 中）被指定下来。

附件 A.1.1 简介

```
<tt:MessageDescription IsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="ProfileToken"
Type="tt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:ElementItemDescription Name="Config"
Type="tt:Profile"/>
</tt:Data>
```

附件 A.1.2 视频源配置

```
<tt:MessageDescription IsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="VideoSourceConfigurationToken"
Type="tt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:ElementItemDescription Name="Config"
Type="tt:VideoSourceConfiguration"/>
</tt:Data>
</tt:MessageDescription>
```

附件 A.1.3 音频源配置

```
<tt:MessageDescription IsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="AudioSourceConfigurationToken"
Type="tt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:ElementItemDescription Name="Config"
```

```
Type="tt:AudioSourceConfiguration"/>
</tt:Data>
</tt:MessageDescription>
```

附件 A.1.4 视频编码配置

```
<tt:MessageDescription iIsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="VideoEncoderConfigurationToken"
Type="tt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:ElementItemDescription Name="Config"
Type="tt:VideoEncoderConfiguration"/>
</tt:Data>
</tt:MessageDescription>
```

附件 A.1.5 音频编码配置

```
<tt:MessageDescription iIsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="AudioEncoderConfigurationToken"
Type="tt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:ElementItemDescription Name="Config"
Type="tt:AudioEncoderConfiguration"/>
</tt:Data>
</tt:MessageDescription>
```

附件 A.1.6 视频分析配置

```
<tt:MessageDescription IsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="VideoAnalyticsConfigurationToken"
Type="tt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:ElementItemDescription Name="Config"
Type="tt:VideoAnalyticsConfiguration"/>
</tt:Data>
</tt:MessageDescription>
```

附件 A.1.7 PTZ 配置

```
<tt:MessageDescription IsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="PTZConfigurationToken"
Type="tt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:ElementItemDescription Name="Config"
```

Type="tt:PTZConfiguration"/>
</tt:Data>
</tt:MessageDescription>

附件 A.1.8 元数据配置

<tt:MessageDescription IsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="MetaDataConfigurationToken"
Type="tt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:ElementItemDescription Name="Config"
Type="tt:MetaDataConfiguration"/>
</tt:Data>
</tt:MessageDescription>

附件 A.1.9 设备管理主题

该设备主题包含以下定义在 ONVIF 主题名字空间中的 Sub-topics

tns1:Device/Trigger/Relay
tns1:Device/OperationMode/ShutdownInitiated
tns1:Device/OperationMode/UploadInitiated
tns1:Device/HardwareFailure/FanFailure
tns1:Device/HardwareFailure/PowerSupplyFailure
tns1:Device/HardwareFailure/StorageFailure
tns1:Device/HardwareFailure/TemperatureCritical

只有重播定义了一个有效载荷，其他的主题用一个空消息来答复。

附件 A.1.10 答复

<tt:MessageDescription IsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="RelayToken" Type="tt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:SimpleItemDescription Name="LogicalState"
Type="tt:RelayLogicalState"/>
</tt:Data>
</tt:MessageDescription>

附件 A.1.11 PTZ 控制器主题

PTZ 服务指定处理 PTZ 预置。由于移动操作是非阻塞的，一个 NVC 将不被通知当已经完成一个 PTZ 预置的时候。因此，介绍了下面的事情（通知用户的状态来预置运动）

tns1:PTZController/PTZPresets/Invoked
tns1:PTZController/PTZPresets/Reached
tns1:PTZController/PTZPresets/Aborted
tns1:PTZController/PTZPresets/Left

典型的事件顺序要求预设一个 NVC 预设一个初始设置。当一个设备接受了这个请求，它会发出一个调用事件，调用事件不得不遵循一个已经完成的事件或者一个已经终止的事件。当球到达调用预置位的时候，前者被使用，后者就在其他的情况下被使用。完成的事件必须遵

从左边的事件，一旦球移动远离了预设点。
这些事件的信息结构被下面的消息描述给出来了：

```
<tt:MessageDescription>
<tt:Source>
<tt:SimpleItemDescription Name="PTZConfigurationToken"
Type="tt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:SimpleItemDescription Name="PresetToken" Type="tt:ReferenceToken"/>
<tt:SimpleItemDescription Name="PresetName" Type="tt:Name"/>
</tt:Data>
</tt:MessageDescription>
```

附件 B(场景描述)

附件 B(场景描述)

B .1 颜色描述符

颜色描述符是作为一个对象节点的外观节点的元素选项被定义的。颜色描述符被定义成了一系列的颜色集（每个组成了一个颜色值，一个可选的重量，一个可选的协方差矩阵）。颜色描述不确定，这个颜色集群是如何创建的。他们可以代表一个一箱颜色直方图或类聚算法的结果。

颜色是由三位向量表示的，此外，每个颜色向量的颜色空间可以被一个颜色空间属性来确定。如果颜色空间向量消失，就假设成了 YCbCr 颜色空间。它指的是整个领域的'sRGB。对于 YCbCr 的颜色空间来说 Colourspace URI 在 is www.onvif.org/ver10/colorspace/YCbCr.上。

```
<xs:complexType name="ColorDescriptor">
<xs:sequence>
<xs:element name="ColorCluster" minOccurs="0" maxOccurs="unbounded">
<xs:complexType>
<xs:sequence>
<xs:element name="Color" type="tt:Color"/>
<xs:element name="Weight" type="xs:float" minOccurs="0"/>
<xs:element name="Covariance" type="tt:ColorCovariance" minOccurs="0"/>
...
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
<xs:complexType name="Color">
<xs:attribute name="X" type="xs:float" use="required"/>
```

```
<xs:attribute name="Y" type="xs:float" use="required"/>
<xs:attribute name="Z" type="xs:float" use="required" />
<xs:attribute name="Colorspace" type="xs:anyURI"/>
</xs:complexType>
<xs:complexType name="ColorCovariance">
<xs:attribute name="XX" type="xs:float" use="required"/>
<xs:attribute name="YY" type="xs:float" use="required"/>
<xs:attribute name="ZZ" type="xs:float" use="required" />
<xs:attribute name="XY" type="xs:float"/>
<xs:attribute name="XZ" type="xs:float"/>
<xs:attribute name="YZ" type="xs:float" />
<xs:attribute name="Colorspace" type="xs:anyURI"/>
</xs:complexType>
```

B1.1 类描述符

一个类描述符是作为一个对象节点的外观节点的可选元素定义的。一个类描述符是被一个类对象的列表和属于这一相应对象的可能性来定义的。

```
<xs:simpleType name="ClassType">
<xs:restriction base="xs:string">
<xs:enumeration value="Animal"/>
<xs:enumeration value="Face"/>
<xs:enumeration value="Human"/>
<xs:enumeration value="Vehicle"/>
<xs:enumeration value="Other"/>
</xs:restriction>
</xs:simpleType>
<xs:complexType name="ClassDescriptor">
<xs:sequence>
<xs:element name="ClassCandidate" minOccurs="0" maxOccurs="unbounded">
<xs:complexType>
<xs:sequence>
<xs:element name="Type" type="tt:ClassType"/>
<xs:element name="Likelihood" type="xs:float"/>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
```

参考数目

[EAP-Registry] Extensible Authentication Protocol (EAP) Registry
[http://www.iana.org/assignments/eap-numbers/eap-numbers.xml]
ONVIF Security Recommendations White Paper
[http://www.onvif.org/portals/3/documents/whitepapers/ONVIF_Security_Recommendations_ver10.pdf]
ONVIF PTZ Coordinate Spaces White Paper

[http://www.onvif.org/Portals/0/documents/whitepapers/ONVIF_PTZ_coordinate_spaces.pdf]
RFC 2396, Uniform Resource Identifiers (URI): Generic Syntax, T. Berners-Lee et al., August 1998
[URL:http://www.ietf.org/rfc/rfc2396.txt]
[UDDI API ver2, “UDDI Version 2.04 API Specification UDDI Committee Specification, 19 July 2002”,
OASIS standard, 19 July 2002
[URL:http://uddi.org/pubs/ProgrammersAPI-V2.04-Published-20020719.pdf]
[UDDI Data Structure ver2] “UDDI Version 2.03 Data Structure Reference UDDI Committee Specification”, OASIS standard, 19 July 2002.
URL:http://uddi.org/pubs/DataStructure-V2.03-Published-20020719.pdf
[WS-KerberosToken] “Web Services Security Kerberos Token Profile 1.1”, OASIS Standard, ,1 February
2006.
URL:http://www.oasis-open.org/committees/download.php/16788/wss-v1.1-spec-os-KerberosTokenProfile.pdf
[WS-SAMLToken] “Web Services Security: SAML Token Profile 1.1”, OASIS Standard, 1 February 2006.
URL:http://www.oasis-open.org/committees/download.php/16768/wss-v1.1-spec-os-SAMLTokenProfile.pdf
[WS-X.509Token] “Web Services Security X.509 Certificate Token Profile 1.1”, OASIS Standard,1 February 2006.
URL:http://www.oasis-open.org/committees/download.php/16785/wss-v1.1-spec-os-x509TokenProfile.pdf
[WS-RELToken] “Web Services Security Rights Expression Language (REL) Token Profile 1.1”, OASIS Standard, 1 February 2006
URL:http://www.oasis-open.org/committees/download.php/16687/oasis-wss-rel-token-profile-1.1.pdf
[X.680] ITU-T Recommendation X.680 (1997) | ISO/IEC 8824-1:1998, Information Technology - Abstract Syntax Notation One (ASN.1): Specification of Basic Notation.
[X.681] ITU-T Recommendation X.681 (1997) | ISO/IEC 8824-2:1998, Information Technology - Abstract Syntax Notation One (ASN.1): Information Object Specification.
[X.682] ITU-T Recommendation X.682 (1997) | ISO/IEC 8824-3:1998, Information Technology - Abstract Syntax Notation One (ASN.1): Constraint Specification.
[X.683] ITU-T Recommendation X.683 (1997) | ISO/IEC 8824-4:1998, Information Technology - Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 Specifications.
[X.690] ITU-T Recommendation X.690 (1997) | ISO/IEC 8825-1:1998, Information Technology - ASN.1 Encoding Rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER).
[ONVIF Analytics WSDL] ONVIF Video Analytics Service WSDL, ver 2.0, 2010.

URL:http://www.onvif.org/onvif/ver10/analytics/wsdl/analytics.wsdl
[ONVIF DM WSDL] ONVIF Device Management Service WSDL, ver 2.0, 2010.
URL:http://www.onvif.org/onvif/ver10/device/wsdl/devicemgmt.wsdl
[ONVIF Event WSDL] ONVIF Event Service WSDL, ver 2.0, 2010.
URL:http://www.onvif.org/onvif/ver10/event/wsdl/event.wsdl
[ONVIF Imaging WSDL] ONVIF Imaging Service WSDL, ver 2.0, 2010.
URL:http://www.onvif.org/onvif/ver10/imaging/wsdl/imaging.wsdl
[ONVIF Media WSDL] ONVIF Media Service WSDL, ver 2.0, 2010.
URL:http://www.onvif.org/onvif/ver10/media/wsdl/media.wsdl
[ONVIF PTZ WSDL] ONVIF PTZ Service WSDL, ver 2.0, 2010.
URL:http://www.onvif.org/onvif/ver10/ptz/wsdl/ptz.wsdl
[ONVIF DP WSDL] ONVIF Remote Discovery Proxy Services WSDL, ver 2.0, 2010.
URL:http://www.onvif.org/onvif/ver10/network/wsdl/remotediscovery.wsdl
[ONVIF Schema] ONVIF Schema, ver 2.0, 2010.
URL:http://www.onvif.org/onvif/ver10/schema/onvif.xsd
[ONVIF Topic Namespace] ONVIF Topic Namespace XML, ver 2.0, 2010.
URL:http://www.onvif.org/onvif/ver10/topics/topicns.xml
WS-I, Basic Profile Version 2.0 – Working Group Draft, C. Ferris (Ed), A. Karmarkar (Ed) and P. Yendluri
(Ed), October 2007.
<http://www.ws-i.org/Profiles/BasicProfile-2_0(WGD).html>