Event Data Streaming.

Dynamic Integration of event producing applications and services
Contents

> What is dynamic Integration?
> How does dynamic integration work?
> The event stream in detail
> Alternatives for receiving events
> Pointers for implementation
What is Dynamic Event Integration?

- The ability to handle events without any prior built-in knowledge about those events
- No more separate .cgi’s for motion detection, audio detection, etc.
- These all imply prior knowledge about their implementation
Why Dynamic Event Integration?

> The list of events supported by a camera doesn’t just depend on the model.

> It also depends on its configuration, like presence of ACAPs.

> So it becomes flexible.

> If you don’t anticipate this, your software is less flexible than the rest of the system.

> It may become a bottleneck.
A VMS typically doesn’t need to know what an event is.

The camera provides all event information needed:
- to show to a human the available events and associated states
- to tell a machine how to subscribe to the event occurrences, and process them

The decision what to do on which event can be entirely given to the human that configures the system.
Dynamic Event Integration in pictures...
Conventional solution: VMS driven integration

Action rule configuration

Event X

TCP/HTTP

VMS

Action rule
Send msg to recipient Y on event X

VMS rule
Record on http message
Solution 2: ACAP/SI driven integration

ACAP

Event X

On X send message to...

TCP/HTTP

IP-address & port

Detector config

VMS

VMS

VMS rule
Record on
http message
Solution 3: Event streaming (Unified architecture)

Event data stream

Camera

Event X

ACAP

Event Y

VMS

Action rule X

Action rule Y

www.axis.com
Dynamic event integration: how to setup

Event declaration list

Query event list

VMS

Action rule
Record video
on event Y

System configuration
Dynamic event integration in ACS

New Rule

Steps
Triggers
Actions
Schedule
Details

Entering a name and click ‘Finish’ to create the rule.

Name: My sunrise rule

Triggers: Device event trigger on Cam 15 (suncalc)
Schedule: Always on
Actions: Raise alarm 'Sunrise'
ACAP configuration
Configuration of ACAP: Conventional solution

ACAP configuration

VMS

Action rule Y
Configuration of ACAP: In VMS context

- Query application info
- Configuration URL
- ACAP
- System configuration
- Operator
- VMS
ACAP configuration without specific knowledge

> **Run:** http://<servername>/axis-cgi/applications/list.cgi

```xml
<reply result="ok">
  <application
    Name="CrossLineDetection"
    NiceName="Cross Line Detection"
    Version="1.0"
    Vendor="Axis Communications"
    Status="Stopped"
    License="missing"
    ApplicationID="3051"
    ConfigurationPage="local/CrossLineDetection/setupCrossLine.shtml"
  ></application>
</reply>
```

> Load that URL in an embedded window
The Event Stream
What is the Event Stream?

> A generic interface to report *all* events
> Events are sent as XML document over RTP/RTSP, as a stream
> Was introduced together with ONVIF in firmware 5.09
> VAPIX was extended to support the same scheme as ONVIF
> It is based on SOAP WebServices
  – However, it is not necessary to have full SOAP support
  – HTTP POST and a bit of XML will do

<?xml version="1.0"?>
<quiz>
  <qanda seq="1">
    <question>
      Who was the forty-second president of the U.S.A.?
    </question>
    <answer>
      William Jefferson Clinton
    </answer>
  </qanda>
<!-- Note: We need to add more questions later.-->
</quiz>
Why RTP/RTSP?

> RTP is a protocol to ‘stream’ data with timestamps

> Using RTP the timing of different streams can be related to each other
  – Typical: audio and video
  – But also: metadata and video

> RTSP is required to setup the RTP connection.
  – The sender and receiver negotiate transport-type and port numbers with each other
Difference with ONVIF

- ONVIF and VAPIX use the same event data stream format.
- ONVIF contains a standardized set of events.
- This is a subset of the camera functions.
- The VAPIX Event Stream contains everything the camera offers.
- More practical: The VAPIX Webservice uses simpler to implement authentication compared to ONVIF.
Documentation

> Documentation on the partner pages

- Event and Actions API
- Event Data Streaming
How does the Event Stream work? Example (part 1 of 4)

First, request the events description:

```xml
<SOAP-ENV:Envelope
    xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
    xmlns:aev="http://www.axis.com/vapix/ws/event1">
  <SOAP-ENV:Body>
    <aev:GetEventInstances />
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```
How does the Event Stream work? Example (part 2 of 4)

Example excerpt of response:

```xml
<SOAP-ENV:Body>
  <aev:GetEventInstancesResponse>
    <wstop:TopicSet>
      ...
      <tns1:Device aev:NiceName="Device">
        <tnsaxis:IO aev:NiceName="Input ports">
          <Port wstop:topic="true" aev:NiceName="Digital input port">
            <aev:MessageInstance aev:isProperty="true">
              <aev:SourceInstance>
                <aev:SimpleItemInstance Name="port" Type="xsd:int" aev:NiceName="Port">
                  <aev:Value aev:NiceName="Any">-1</aev:Value>
                  <aev:Value aev:NiceName="Input 1">0</aev:Value>
                </aev:SimpleItemInstance>
              </aev:SourceInstance>
              <aev:DataInstance>
                <aev:SimpleItemInstance Name="state" Type="xsd:boolean" isPropertyState="true" aev:NiceName="Active" />
              </aev:DataInstance>
            </aev:MessageInstance>
          </Port>
        </tnsaxis:IO>
      </tns1:Device>
    </wstop:TopicSet>
  </aev:GetEventInstancesResponse>
</SOAP-ENV:Body>
```
To subscribe to a stream that reports Digital input 1 state changes:

rtsp://a.b.c.d/axis-media/media.amp?video=0&audio=0&event=on&
eventtopic=onvif:Device/axis:IO/Port&
eventcontent=boolean(//SimpleItem[@Name="port" and @Value="1"])
How does the Event Stream work? Example (part 4 of 4)

> In the stream you will receive messages like:

```xml
<tt:MetadataStream>
  <tt:Event>
    <wsnt:NotificationMessage>
      <wsnt:Topic Dialect="http://docs.oasis-open.org/wsn/t1/TopicExpression/Simple">
        tns1:Device/tnsaxis:IO/Port
      </wsnt:Topic>
        <tt:Source>
          <tt:SimpleItem Name="port" Value="0" />
        </tt:Source>
        <tt:Key />
        <tt:Data>
          <tt:SimpleItem Name="state" Value="1" />
        </tt:Data>
      </wsnt:Message>
    </wsnt:NotificationMessage>
  </tt:Event>
</tt:MetadataStream>
```
Our documentation sometimes formats " as “ or ”. Copy-paste of this content gives errors

Escaping only needs to be performed on the characters: =, " and space

The set of events may change. The VMS shouldn’t cache event declarations

Pre 5.50 firmware has small ‘deviations’. Take care to:
  – not use fixed eventtopics. Build them from the GetEventInstances output
  – not use eventcontent expressions from pre-5.50 firmware.

Obviously, GetEventInstances-based integration doesn’t work for offline configuration
Alternatives: other ways to receive camera events

> Triggerdata/Userdata
  - Found in JPEGs and in RTP stream
  - Pro: Lightweight and simple
  - Con: Fixed set of information
  - See: Partner pages

> ONVIF
  - A similar streaming mechanism is present
  - Pro: 100% standard
  - Con: limited to what is standardized
How to receive the Event Stream?

> If you already do your own RTSP, it’s likely to be a small step

> If your RTSP is heavily tied to the video path (like AMC), you have a challenge

> A few slides to provide pointers on how to do this
Use an existing client: C#

Note: we have not tested these. These look reasonable and may save you time

> Commercial solutions available, starting at €2450

> Managed Media Aggregation (open source)
Use existing clients: C/C++

> Gstreamer
  - Platform independent
  - Complete media framework
  - Available for many languages:  
  - Can be used with Windows/C#
Live555

- RTSP client used in VLC
- C++
- Supported on many platforms
- Example available
- Con: ‘Forces’ you in the live555 way
- Pro: Takes care of RTP interleaving
- Compiling live555 with modern Visual Studio:
  You need this workaround
Use existing clients: C/C++

> libcurl

- Plain C interface
- Supported on many platforms
- Pro: More flexible than live555
- Con: RTSP support is limited
- Con: Needs patches
- Example available
Do-it-yourself Clients

> There are several means of transport
  - RTSP + RTP over UDP
  - RTSP + RTP over RTSP
  - RTSP + RTP over RTSP over HTTP

> And several standards involved
  - RTSP
  - SDP
  - RTP, RTP interleaving
  - HTTP
  - XML
Do-it-yourself Clients

> With a bit of simplification, it isn’t so hard
>
> Consider RTP over RTSP over HTTP.
>  – This gets you through firewalls, proxies, SSL, …
>  – Use a real HTTP client for this purpose
>
> RSTP over HTTP is defined by Apple
> See http://www.opensource.apple.com/
>
> Two HTTP connections:
>  – A never-ending POST on which you send RTSP commands, base64 encoded
>  – A never-ending GET on which you receive RTSP responses and RTP data
Summary

> Event data is a uniform mechanism to have access to all event data

> There is not a single technology standing out as obvious choice for implementing a receiver

> Remember, our support covers Axis specific questions and not ‘common knowledge’ about programming and protocols

> Still, when you are going in this direction we can share with you what we know

> Contact us