

VCAsys Metadata Format

Contents

Introduction	3
Metadata Streaming	4
Restrictions	4
Encrypted Metadata	4
VCA Meta Render DLL	5
Fundamentals	6
Co-ordinate System	6
Schema (.XSD) File	6
Top Level Schema	7
<vca>/<vca_hdr>	10
<vca>/<vca_cfg>	11
<cfg_update>	12
<time_zone>	12
<vca>/<objects>	13
<bb>	14
<vca>/<events>	15
<vcaTime>	16
<vca>/<counts>	17
<base64map>	18
<vca>/<blobmap>	19
<vca>/<statblobmap>	20
<vca>/<tampermap>	21
<vca>/<scnchgmap>	22
<vca>/<smokemap>	23
<vca>/<firemap>	24
<vca>/<cntlninfo>	25
<ce>	25
<vca>/<stats>	26
<waiting>	26
<wait>	26
Revision History	27

Introduction

This document describes the metadata format for the VCAsys tracking engine. It explains how to integrate the output of the VCAsys tracking engine into a third party product. It covers both the NVC/IPE and IPN/IPX product lines. Where the metadata format corresponds to features which are currently supported only on the NVC/IPE product line, this will be indicated in the description.

Metadata Streaming

It is possible to retrieve the VCA tracking metadata via a number of methods including HTTP and RTSP. See the HTTP API specification for more information about the specifics of requesting and receiving a metadata stream.

It is advisable to use a TCP based protocol for the reception of metadata since it is more vulnerable should packets be dropped by the communication medium. For this reason, it is not advisable to stream metadata over a PCVCA/multicast connection.

Restrictions

Encrypted Metadata

In the most basic version of VCAsys, VCAPresence, sections of metadata may be encrypted. The sections of encrypted metadata are chosen at random. This means that the metadata cannot be used in this version of VCAsys.

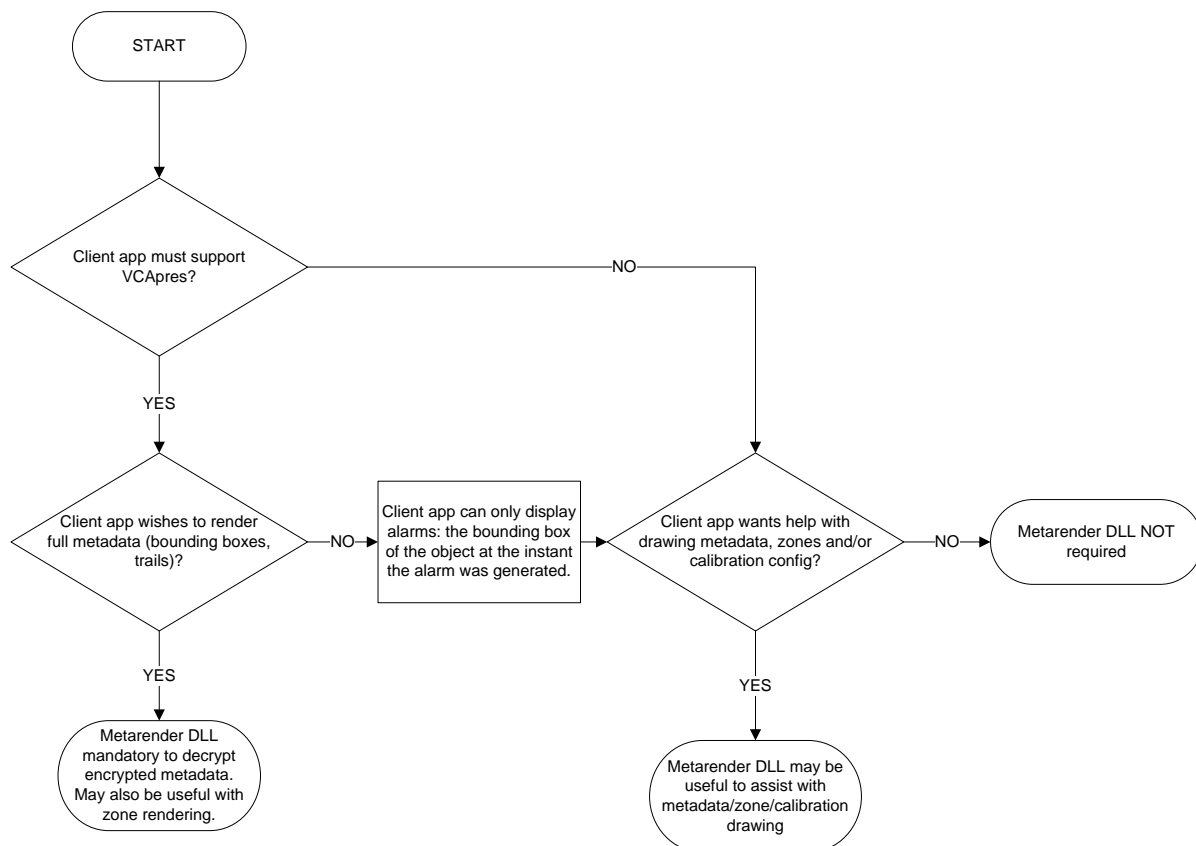
If there is a requirement to use the metadata, it is possible to upgrade the VCAsys engine with a different activation key. This disables the metadata encryption such that it is available for client-side use.

In order to render all of the metadata when using VCAsys, the VCA Meta Render DLL must be used.

VCA Meta Render DLL

The VCA Meta Render DLL is a helper library that can be used by the client application to render data relevant to VCAsys on the screen. The type of data the DLL can render includes object bounding boxes, trails, zones, counters, etc.

In order to decide whether the VCA Meta Render DLL should be used by your application, follow the decision tree, below:



Fundamentals

The VCA tracking metadata is sent as plain-text XML.

The metadata is formatted as an XML document. A complete document is sent each frame.

The VCA metadata consists of the following:

- The status of the VCA processing on the current frame.
- The bounding boxes and trails of all currently tracked objects.
- Notifications of any events that have been generated due to objects triggering rules.
- Notifications of the value of any configured counters.
- The pixel-map of triggered pixels.

It is possible to control the content of the metadata stream. For example, if the client application wishes to receive alarm notifications, but is not interested in the blob map, this can be configured. Please refer to the HTTP API manual (for IP based engine variants), or the SDK manual (for PC/embedded engine variants).

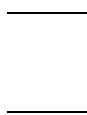
Co-ordinate System

Where co-ordinates are transmitted (e.g, x-y locations of bounding boxes, trail points, etc), these are normalized across 16 bits. i.e, 0-65535 corresponds to 0-100% and are relative to the top left of the image.

Example:

Bounding box: x[25577] y[45510] w[3641] h[2276]

To display this on top of a PAL-D1 (720x576) image, it is de-normalized as follows:

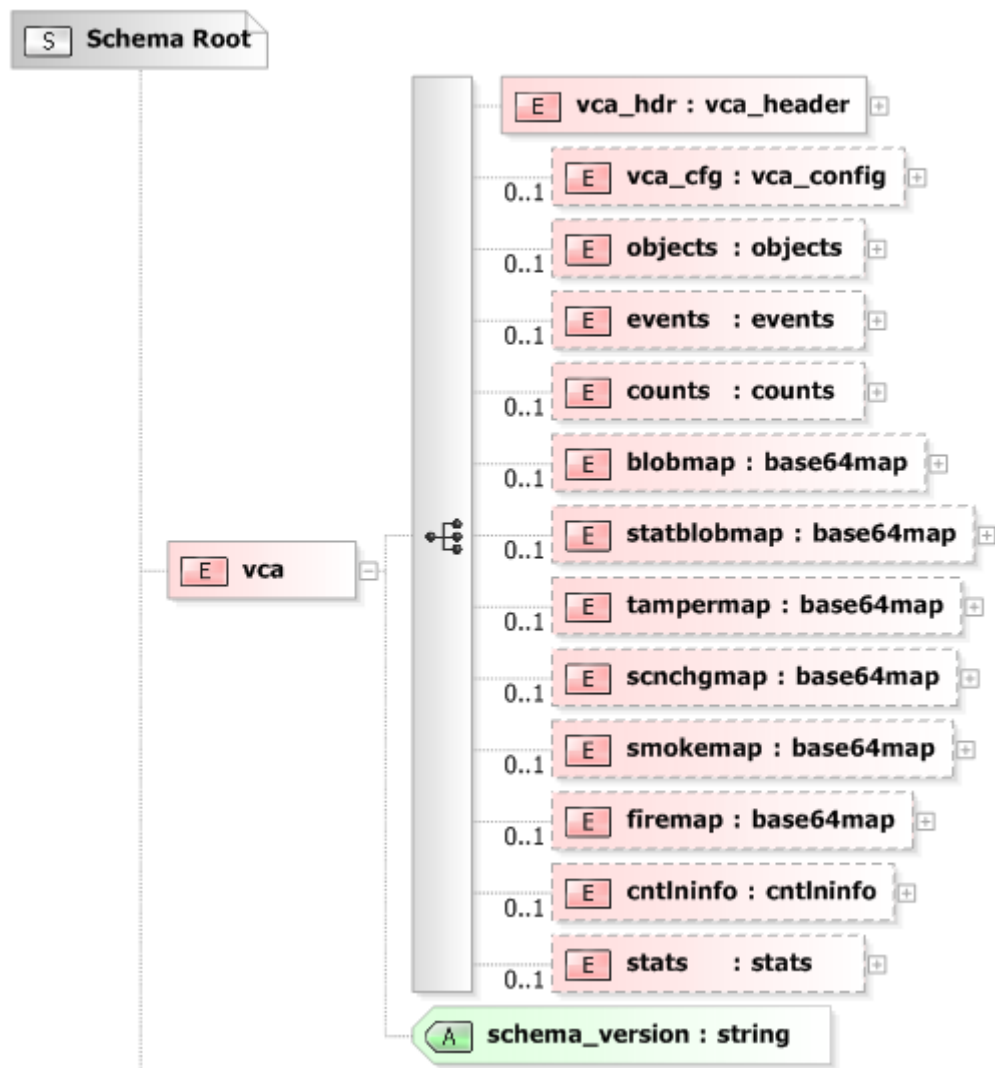


And hence, the final coordinates would be x[281] y[399] w[40] h[20]. Please note: the actual x coordinate range and y coordinate range are [0,719] and [0,575] respectively.

Schema (XSD) File

An XSD schema file is available for validation of the XML metadata, if required. The schema file should be distributed with this document. If not, please contact your vendor.

Top Level Schema



The first levels of the XML schema are as follows:

<vca>

The base element. All metadata is contained within this element.

<vca>/<schema_version>

The attribute which contains the version of metadata schema.

<vca>/<vca_hdr>

The VCA header. Contains information regarding the status of the currently processed frame.

<vca>/<vca_cfg>

VCA configurations. Contains information regarding the configurations including measurement units.

<vca>/<objects>

Object notifications. Contains the bounding boxes and trails of every object that is currently being tracked.

<vca>/<events>

Event notifications. Contains information about all currently active events.

<vca>/<counts>

Counter notifications. Contains updates to the value of any counters that have been configured.

<vca>/<blobmap>

Object detection output data. Contains the output from the object detection stage of the algorithm. Useful for analyzing the performance of the VCA algorithm and when evaluating its suitability for a particular detection scenario.

<vca>/<statblobmap>

Stationary object detection output data. Contains the output from the stationary object detection module. Useful for analyzing the performance of the VCA algorithm and when evaluating its suitability for a particular detection scenario. *(NVC/IPE only)*

<vca>/<tampermap>

Tamper detection output data. Contains the output from the tamper detection module. Useful for analyzing the performance of the VCA algorithm and when evaluating its suitability for a particular detection scenario.

<vca>/<scnchgmap>

Scene change detection output data. Contains the output from the scene change detection module. Useful for analyzing the performance of the VCA algorithm and when evaluating its suitability for a particular detection scenario.

<vca>/<smokemap>

Smoke detection output data. Contains the output from the smoke detection module. Useful for analyzing the performance of the VCA algorithm and when evaluating its suitability for a particular detection scenario. *(NVC/IPE only)*

<vca>/<firemap>

Fire detection output data. Contains the output from the fire detection module. Useful for analyzing the performance of the VCA algorithm and when evaluating its suitability for a particular detection scenario. *(NVC/IPE only)*

<vca>/<cntlninfo>

Counting line notifications. Contains the output of the counting line module.

<vca>/<stats>

Statistics notifications. Contains the output of all the statistics information. ***(NVC/IPE only)***

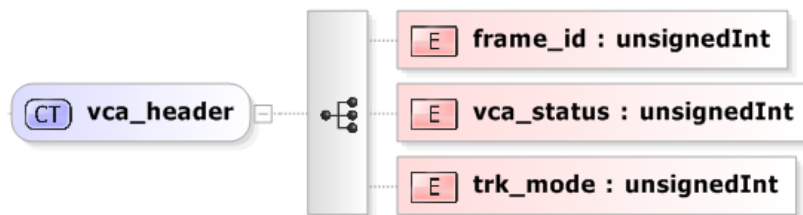
Note: The base elements described above only appear in the XML output stream if processing that generates those elements has occurred in that frame. Since the VCA engine tries to load-balance, it may not generate output elements every frame. If an output element is not specified, then it means that processing has not occurred during that frame to generate those element(s).

<vca>/<vca_hdr>

The <vca_hdr> is sent every frame. It contains information about the general VCA processing status:

Support

- NVC/IPE
- IPN/IPX



- <frame_id>: The id of the frame to which the metadata corresponds. Increments each frame and starts at 0 when the board is reset.
- <vca_status>: The status of the VCA engine while processing the current frame. Values have the following meanings:

Value	Meaning
0	No error
1	Codec not configured
2	Unsupported input format
3	Codec is not licensed
4	Codec is currently learning the scene
5	Codec has been manually inhibited (tracking has been prevented by an external command).

- <trk_mode>: The running mode of VCA Engine::

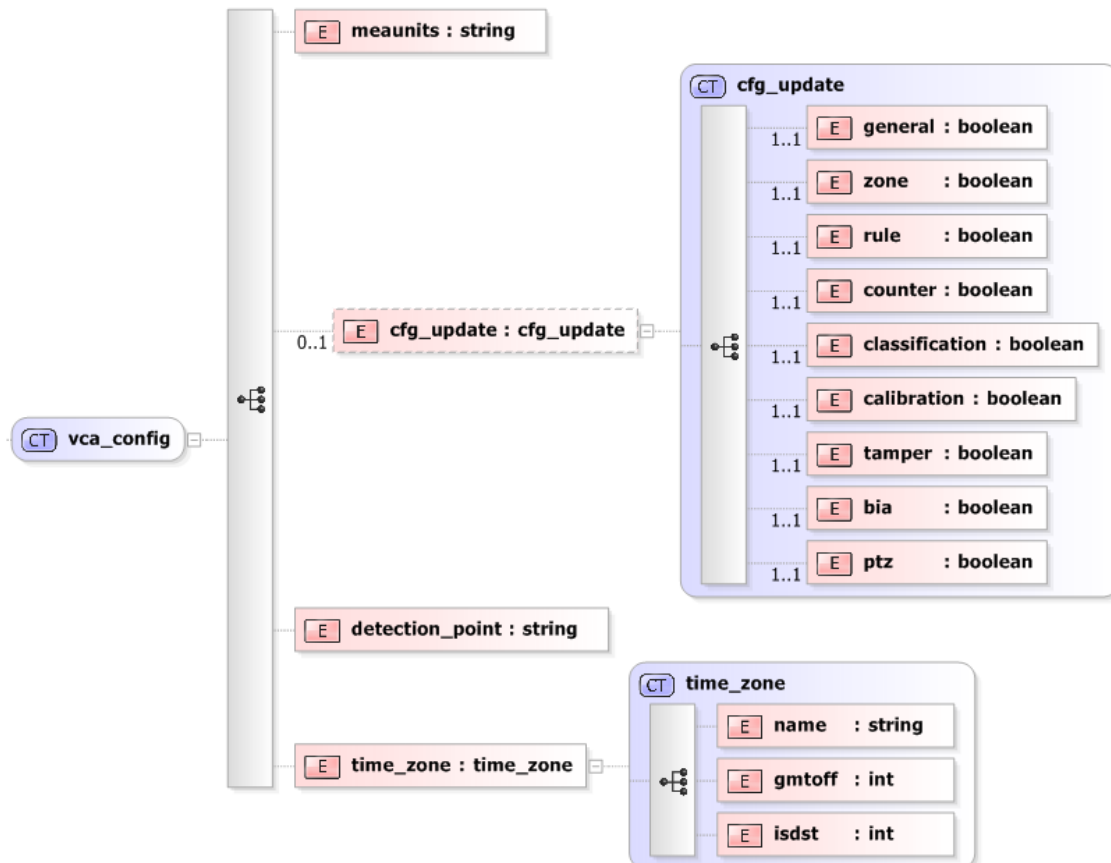
Value	Meaning
0	Fixed camera tracking mode
1	PTZ camera auto-tracking mode

<vca>/<vca_cfg>

The <vca_cfg> is sent every second. It contains information about the general VCA configurations:

Support

- NVC/IPE
- IPN/IPX



- <meaunits>: Type of the measurements units:

Value	Meaning
metric	Metric systems
imperial	Imperial systems

- <cfg_update>: Type of VCA configuration updates
- <detection_point>: Type of detection point for VCA tracked objects:

Value	Meaning
centroid	Centroid as detection point
midbottom	Mid-bottom as detection point

- <time_zone>: Time zone information

<cfg_update>

<cfg_update> is sent when any VCA configuration changes happened. Therefore, if <vca_cfg> element contains no <cfg_update> element, this indicates that there is no VCA configuration update.

- <general>: Boolean value, true if any general configuration changed
- <zone>: Boolean value, true if any detection zone/line configuration changed
- <rule>: Boolean value, true if any detection rule configuration changed
- <counter>: Boolean value, true if any on screen counter configuration changed
- <classification>: Boolean value, true if any object classification configuration changed
- <calibration>: Boolean value, true if any 3D calibration configuration changed
- <tamper>: Boolean value, true if any camera tamper detection configuration changed
- <bia>: Boolean value, true if any burnt-in-annotation configuration changed
- <ptz>: Boolean value, true if any PTZ tracking configuration changed

<time_zone>

<time_zone> represents the time zone information for the VCA system is running:

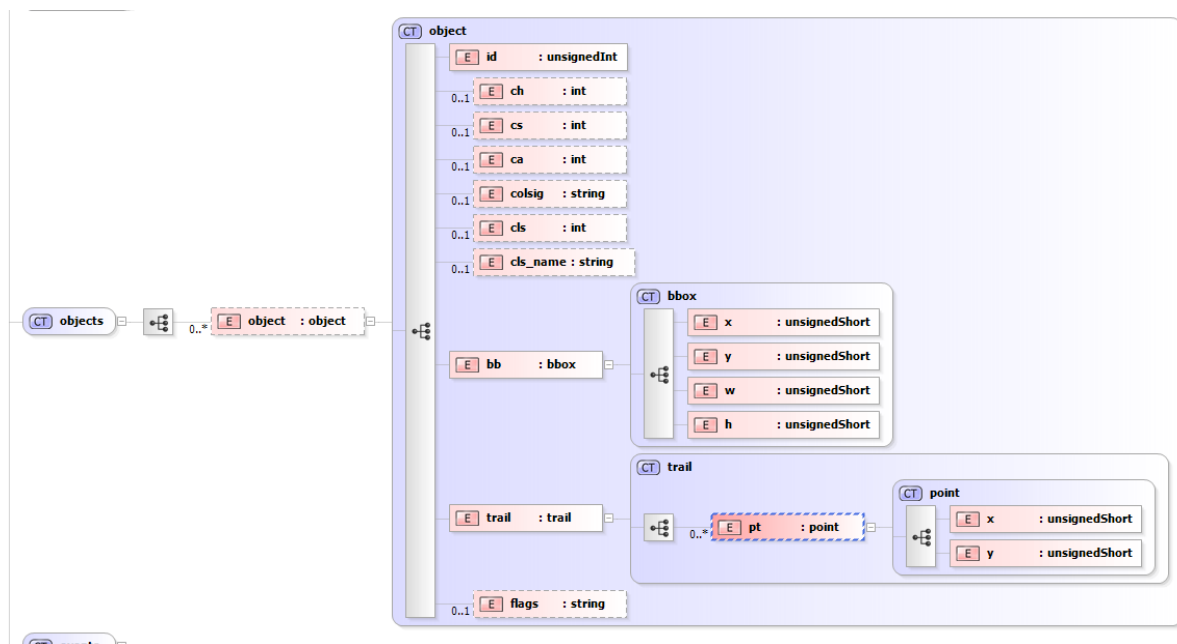
- <name>: The name of time zone
- <gmtoff>: Seconds east of UTC
- <isdst>: A flag that indicates whether daylight saving time is in effect at the time described. The value is positive if daylight saving is in effect, zero if it is not, and negative if the information is not available.

<vca>/<objects>

The <objects> element contains the tracking data for all objects that are currently being tracked by the engine. Similar to other top-level elements, this element is only present if the engine has processed the objects in that frame.

Support

- NVC/IPE
- IPN/IPX



Each <object> contains the following elements:

- <id>: The id of the object.
- <ch>: Calibrated height. The real object height in real-world units according to the current calibration settings.
- <cs>: Calibrated speed. The real object speed in real-world units according to the current calibration settings.
- <ca>: Calibrated area. The real object area in real-world units according to the current calibration settings.
- <colsig>: Colour signature. A string comprising 10 hexadecimal numbers representing the portion of pixels in the tracked object corresponding to each of 10 different colours. Each number has a value between 0 and 255. In order, the colours are black, brown, grey, blue, green, cyan, red, magenta, yellow and white. Note that the colour signature is only

available when the people tracking mode is selected. *(NVC only)*

- **<bb>**: The bounding box of the object being tracked at the current position.
- **<trail>**: The trail of the object being tracked.
- **<cls>**: The classification id of the object. Depends on the classification classes configured on the unit. These can be retrieved from the API. Special reserved values have the following meanings:

Value	Status
-1	Unclassified (does not fit into any classification class)
-2	Unknown (the engine is still analyzing the object parameters and determining the best classification)

- **<cls_name>**: The classification name of the object.
- **<flags>**: Reserved for future use.

<bb>

A **<bb>** is a bounding box of an object. It contains the following elements:

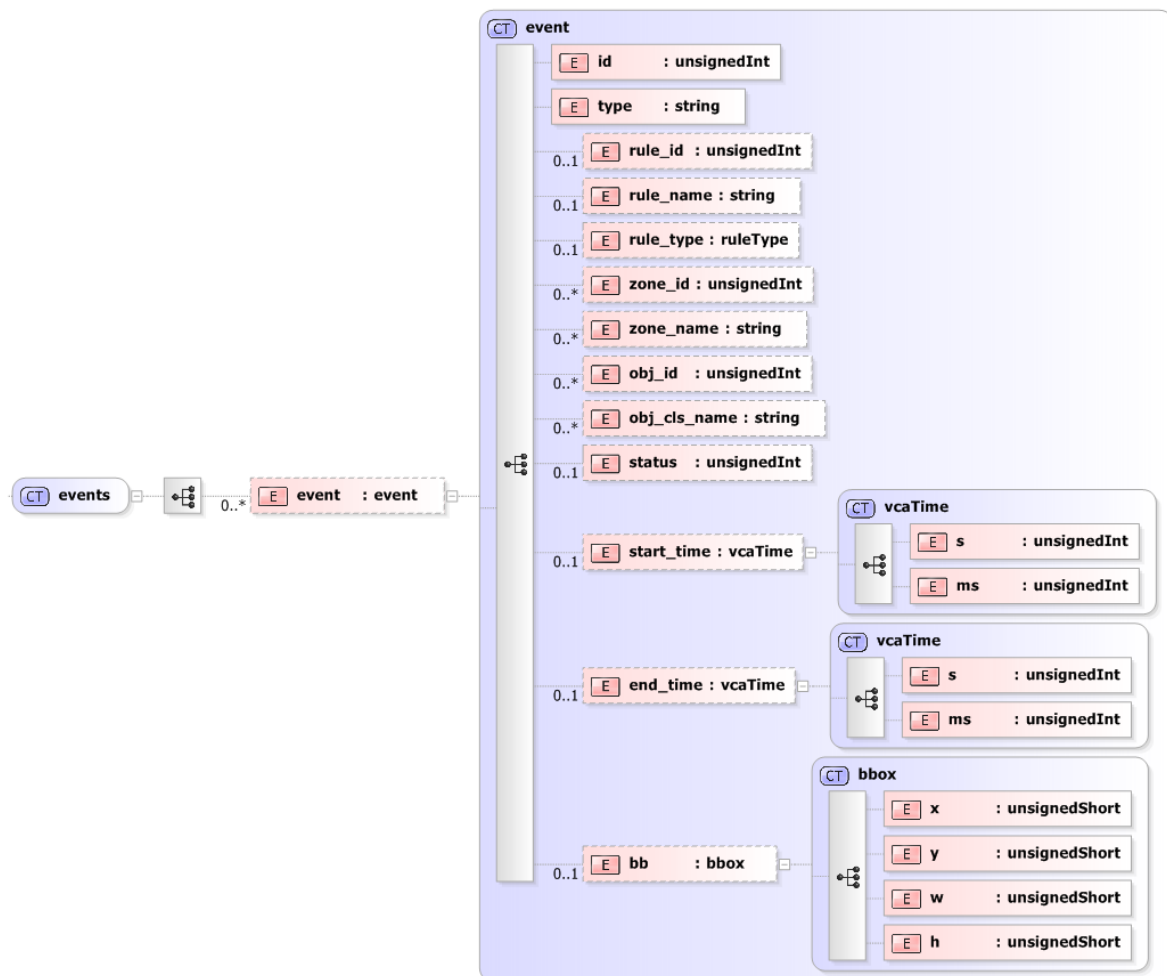
- **<x>**: The normalized x coordinate of the centre of the bounding box.
- **<y>**: The normalized y coordinate of the centre of the bounding box.
- **<w>**: The normalized width of the bounding box.
- **<h>**: The normalized height of the bounding box.

<vca>/<events>

The <events> element is sent every time event processing has been performed by the VCA engine. Therefore, if an <events> element contains no <event> elements, this indicates that event analysis has been performed, but no objects have triggered any rules.

Support

- NVC/IPE
- IPN/IPX



Each <event> contains the following elements:

- <id>: The id of the event.
- <type>: Type of event. Can be:

Value	Meaning
zone	Alarm triggered by zone rule
tamper	Alarm triggered by tamper detection module

NOTE: If <type>='tamper' then the only <id> and <start_time> tags will be present.

- <rule_id>: The id of the rule that triggered to generate the event.
- <rule_name>: The name of the rule that triggered to generate the event.
- <rule_type>: The type of the rule that triggered to generate the event: presence, enter, exit, appear, disappear, stop, dwell, direction, speed, tailgating, linecountera, linecounterb, colsig, smoke, fire, abrmobj, unknown.
- <zone_id>: Zone ids of zones that caused the event to be generated.
- <zone_name>: Zone names of zones that caused the event to be generated.
- <obj_id>: Object ids of the objects that caused the event to be generated.
- <obj_cls_name>: The classification names of the objects that caused the event to be generated.
- <bb>: The bounding box of the object when the event was generated.
- <status>: Alarm status. Can be one of the following:

Value	Status
1	Alarm started
2	Alarm update
3	Alarm ended
4	Pulsed event (the event started and stopped within the same frame)

- <start_time>: The time at which the event was started.
- <end_time>: The time at which the event expired.

<vcaTime>

<vcaTime> represents the elapsed time since 00:00 hours, Jan 1, 1970 UTC. It contains the following elements:

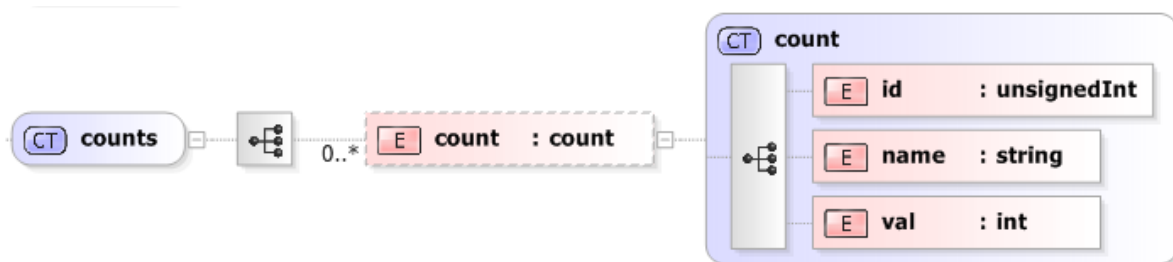
- <s>: The number of whole seconds of elapsed time.
- <ms>: The rest of the elapsed time (a fraction of a second), represented as the number of milliseconds, always less than 1000.

<vca>/<counts>

The <counts> element contains the current value of all counters configured in the engine. Similar to all other elements, this element is only present if the engine has processed the counters in that frame.

Support

- NVC/IPE
- IPN/IPX

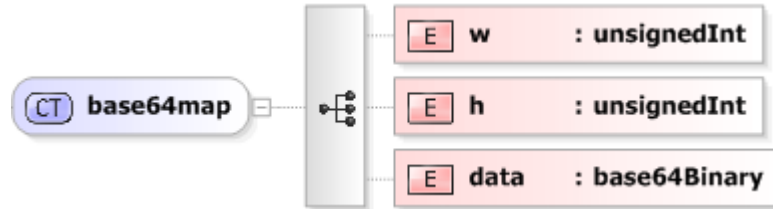


Each <count> consists of the following elements:

- <id>: The id of the counter.
- <name>: The name of the counter.
- <val>: The current value of the counter.

<base64map>

<base64map> contains the binary data representing the fired pixels from the specific VCA algorithm. This data can be helpful when evaluating the suitability of VCA algorithm for a particular application.



The <blobmap> element contains the following elements:

- <w>: The width, in pixels of the blobmap.
- <h>: The height, in pixels of the blobmap.
- <data>: The binary data, in base64 format corresponding to the pixel blob map. The data should be interpreted according to the format element. Normally the size of blobmap data is quite big (176x144/6=4224 Bytes for NVC1000 PAL standard). Please refer to Wikipedia for detailed information regarding base64: <http://en.wikipedia.org/wiki/Base64>.

The Base64 index table is like this:

Binary	ASCII	Binary	ASCII	Binary	ASCII	Binary	ASCII
000000	A	010000	Q	100000	g	110000	w
000001	B	010001	R	100001	h	110001	x
000010	C	010010	S	100010	i	110010	y
000011	D	010011	T	100011	j	110011	z
000100	E	010100	U	100100	k	110100	0
000101	F	010101	V	100101	l	110101	1
000110	G	010110	W	100110	m	110110	2
000111	H	010111	X	100111	n	110111	3
001000	I	011000	Y	101000	o	111000	4
001001	J	011001	Z	101001	p	111001	5
001010	K	011010	a	101010	q	111010	6
001011	L	011011	b	101011	r	111011	7
001100	M	011100	c	101100	s	111100	8
001101	N	011101	d	101101	t	111101	9
001110	O	011110	e	101110	u	111110	+
001111	P	011111	f	101111	v	111111	/

As you can see from the table, each ASCII value stands for a 6-bit binary value and each bit stands for 1 pixel. For example L stands for 6 pixels like this:



<vca>/<blobmap>

The <blobmap> element contains the binary data representing the fired pixels from the object segmentation stage of the tracking engine. This data can be helpful when evaluating the suitability of VCA for a particular application. Refer to <base64map> for more information.

Support

- NVC/IPE
- IPN/IPX

<vca>/<statblobmap>

The <statblobmap> element contains the binary data representing the fired pixels from the stationary object detection algorithm. Refer to <base64map> for more information.

Support

- NVC/IPE

<vca>/<tampermap>

The <tampermap> element contains details of the state of each tamper block. The tamper detection module splits the image into blocks. When more than a given number of blocks are triggered, the tamper alarm is raised. Refer to <base64map> for more information.

Support

- NVC/IPE
- IPN/IPX

<vca>/<scnchgmap>

The <scnchgmap> element contains details of the state of each scene change block when using the manual scene change mode. The detection module splits the image into blocks. When more than a number of blocks are triggered, the scene is determined to have changed, the tracking algorithm is reset and learns the new scene. Refer to <base64map> for more information.

Support

- NVC/IPE
- IPN/IPX

<vca>/<smokemap>

The <statblobmap> element contains the binary data representing the fired pixels from the smoke detection algorithm. Refer to <base64map> for more information.

Support

- NVC/IPE

<vca>/<firemap>

The <statblobmap> element contains the binary data representing the fired pixels from the fire detection algorithm. Refer to <base64map> for more information.

Support

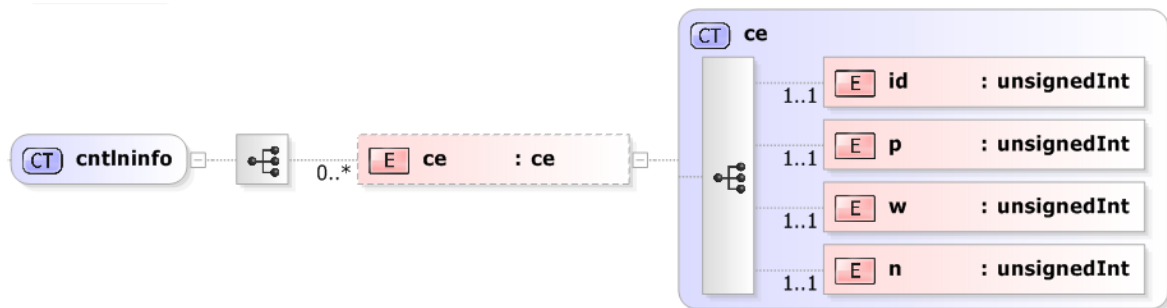
- NVC/IPE

<vca>/<cntlninfo>

The <cntlninfo> element contains information about the output of the counting line module. This information can be useful when configuring the counting lines. It is only sent when this information is generated.

Support

- NVC/IPE
- IPN/IPX



The <cntlninfo> element contains the following elements:

- <ce>: Counting line event

There can be an unlimited number of <ce> elements in the <cntlninfo> element.

<ce>

The <ce> element contains details of a single counting line event. It contains the following elements:

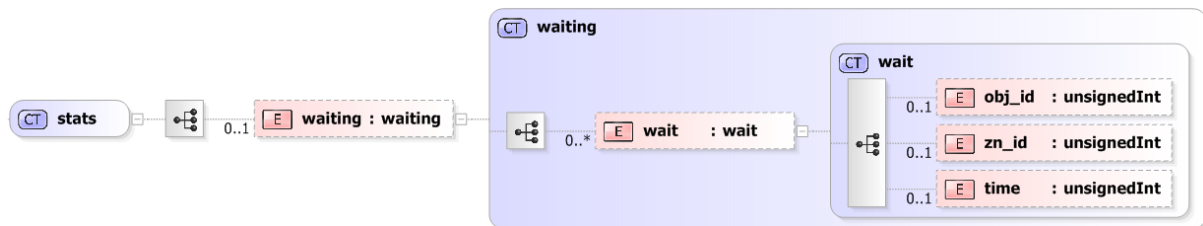
- <id>: The rule ID which the event is associated with.
- <p>: The normalised (65535/screen width) start position of the event from the starting point of the counting line.
- <q>: The normalised (65535/screen width) of the event.
- <n>: The number of counts that the event represents.

<vca>/<stats>

The <stats> element contains all information about the statistics. Currently it only contains the waiting time information.

Support

- NVC/IPE



The <stats> element contains the following elements:

- <waiting>: Overall waiting time information

<waiting>

The <waiting> element contains details of waiting information. It contains the following elements:

- <wait>: Individual waiting information for each object corresponding to each individual zone.

There can be an unlimited number of <wait> elements in the <waiting> element.

<wait>

The <wait> element contains details of individual waiting information. It contains the following elements:

- <obj_id>: The object ID for waiting time information.
- <zn_id>: The zone ID for waiting time information.
- <time>: The actual time in seconds for the object waiting inside the zone

Revision History

Version	Description	Pages Affected
1.20	First documented version	All
1.30	Added <type> tag to <event> structure	8
1.40	Added <rule_name>, <zone_name>, <obj_cls_name> to <event> structure Added <cls_name> to <object> structure Added <name> to <count> structure	8,9,10,12
1.50	Added decision tree regarding VCA Meta Render DLL. Added comment regarding schema XSD file.	5
1.60	Removed erroneous link to "Forensic metadata" document.	6
1.70	Added <tamperinfo>	16
1.80	Added <cntlninfo>	18
1.90	Added <vca_cfg> Added new attribute <schema_version> under <vca> Updated all graphs	All
1.91	Added more descriptions for bloblmap and tampermap	19,20,21
2.00	Added <trk_mode> under <vca_hdr> Added <stats> for statistics information	9,10,22
2.01	Added description for <vcaTime>	13
2.02	Updated <tampermap> (removed <tamperinfo>) Added <scnchgmap>	18,19
2.03	Added status 4 (pulsed event) to event status info table	13
2.04	Added colsig to object descriptor	15
2.05	Added <cfg_update>, <detection_point> and <time_zone> under <vca_cfg> Added <rule_type> under <events>	11,12,14
2.06	Modified document to reference both NVC/IPE and IPN/IPX product lines	All
2.07	Updated and reordered top level schema with statblobmap. Updated and edited the description of the top level schema entries. Updated all diagrams to remove dependency from top-level schema. Reordered sections to be consistent with top-level schema ordering. Added statblobmap entry and edited all blobmap entries.	All
2.08	Added smokemap and firemap entries Added common data type called base64map for blobmap, statblobmap, tampermap, scnchgmap, smokemap and firemap.	All
2.09	Fixed some typos	21
02-2017-A	Manual name changed(VCAsys Metadata Format	1

	Developer's Guide -> VCAsys Metadata Format)	
--	--	--