



Programmer's Manual

VisionWeb Control Reference

Cieffe ActiveX SDK Reference Manual

Created: 2003-05-28 by:
Last modified: 2007-11-05 by: Revision: 419

COPYRIGHT

Copyright © 2003-2007 Insignis Technologies. All rights reserved worldwide.

This manual is the intellectual property of **Insignis Technologies** and is protected by copyright. No part of this document may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photo copying, recording or otherwise without the prior written permission of Insignis Technologies.

The information and intellectual property contained herein is confidential between Insignis Technologies and the client and remains the exclusive property of Insignis Technologies.

TRADEMARKS

Insignis Technologies, Cieffe, Spectiva, Linearis, Proxima, Nettuno, DeltaWavelet, Enpacta, VisionPocket and DeePath names and logos are registered. Other product and company names mentioned herein may be trademarks or registered trademarks of their respective owners.

IMPORTANT NOTICE

Insignis Technologies assumes no liability for any error in this publication and for damages, whether direct, indirect, incidental, consequential or otherwise, that may result from such error, including, but not limited to, loss of data or profits.

Insignis Technologies provides this publication “as is” without warranty of any kind, either expressed or implied, including, but not limited to implied warranties of merchantability or fitness for a particular purpose.

Due to continued product development this published information is subject to change without notice. Insignis Technologies reserves the right to make changes in hardware, firmware, software, user interfaces and layout without notification to its users.

If you find any problems in this manual, please report them to:

Insignis Technologies s.r.l.

Via Laboratori Autobianchi, 1
Edificio 23
20033 Desio . Milano . ITALY

Web Site: www.cieffe.com
E-mail: info@cieffe.com



This page intentionally left blank



Table of contents

Introduction.....	6
Overview.....	6
About this manual.....	6
About ActiveX technology.....	6
About SpectivaWeb, LinearisWeb and ProximaWeb ActiveX controls.....	7
Real-Time View and control of cameras.....	7
Remote management of configuration of alarms.....	7
Recording and transmission of Audio Channels.....	7
Tutorial.....	9
Overview.....	9
Hands on – Borland Delphi and C++ Builder.....	9
Importing the ActiveX Control.....	9
Hands on – Javascript.....	10
Hosting the ActiveX.....	10
Hands on – Microsoft Visual Basic 6.....	11
Importing the ActiveX Control.....	11
Hands on – Microsoft Visual C++ environment.....	12
Importing the ActiveX Control.....	12
Reference.....	14
Overview.....	14
METHODS.....	15
CONNECTION Methods.....	20
WINDOWS MANAGEMENT Methods.....	43
PLAYBACK Methods.....	62
AUX AND ALARMS Methods.....	91
AUDIO MANAGEMENT Methods.....	97
DOME PILOTING Methods.....	100
SPOT MONITOR Methods.....	109
EXPORT Methods.....	115
TEXT INSERTION Methods.....	122
MANUAL RECORDING Methods.....	130
PANORAMA VIEWS Methods.....	135





Introduction

Overview

This chapter provides a general introduction to this manual, its structure and how to use the information within it. It also provides information about:

- general informations about ActiveX technology
- the SpectivaWeb, LinearisWeb and ProximaWeb ActiveX controls
- concepts related to these ActiveX controls

About this manual

This manual is intended for the software developers who will take advantages of the software components developed and provided by Insignis Technologies to interact with its state-of-the-art DVMS client-server systems.

This document is divided into three sections:

1. Introduction
2. Programming guide
3. Reference

The Programming Guide section contains general informations about SpectivaWeb, LinearisWeb and ProximaWeb ActiveX controls and in-depth examples on how to use it to perform specific tasks.

The Reference section lists all the properties and functions exposed by the ActiveX controls, grouped by logical functionalities.

About ActiveX technology

ActiveX controls, formerly known as OLE controls, let you develop sophisticated controls based on the Common Object Model (COM) that can be installed in Windows forms or any ActiveX control container application, including pages on the Internet's World Wide Web and Visual Basic applications.

These controls can be developed for many uses, such as database access, data monitoring, or graphing. Besides their portability, ActiveX controls support features previously not available to custom controls, such as compatibility with existing OLE containers and the ability to integrate their menus with the OLE container menus. In addition, an ActiveX control fully supports Automation, which allows the control to expose writable properties and a set of methods that can be called by



the control user.

An ActiveX control can be embedded in third party applications developed with common software environments that support ActiveX technology: Microsoft Internet Explorer, Microsoft Visual C++, Microsoft Visual Basic, Borland Delphi, Borland C++ Builder, and possibly other.

About SpectivaWeb, LinearisWeb and ProximaWeb ActiveX controls

SpectivaWeb, LinearisWeb and ProximaWeb are two fully customizable ActiveX components that can interface with CIEFFE servers and can be embedded in third party applications developed with common software environments that support ActiveX technology: Microsoft Internet explorer, Microsoft Visual C++, Microsoft Visual Basic, Borland Delphi, Borland C++ Builder, Etc.

Real-Time View and control of cameras

These ActiveX controls make it possible to view simultaneously real-time video streams originating from many cameras connected to many different CIEFFE servers in real time. They dynamically manage connections and disconnections so that the views of cameras can be changed on the fly with little or no delay. They also present a simple interface which allows random access (by time, date, and camera) to all recorded video streams.

If PTZ cameras are connected, each accessed camera can be fully controlled for rapid creation of customized programs, allowing for powerful and flexible automation of entire surveillance systems requiring absolutely no human intervention.

ActiveX controls make use of the rights assignments available on the connected servers so that user access levels can be maintained at all times.

Remote management of configuration of alarms

When an alarm is triggered on an ActiveX enabled server, event notification is sent to the ActiveX application so that appropriate actions can be invoked in response to these, in real time.

Built-in Alarm management and notification functionality allows for powerful integration between CIEFFE DVMS servers and third-party systems via our ActiveX.

Recording and transmission of Audio Channels

CIEFFE DVMS are equipped with powerful audio recording and transmission features with a built-in audio/video mixer (up to 4 audio channels per each video input).

ActiveX controls allows simple creation of audio/video enabled Windows applications only requiring a display device and a sound card to receive live and recorded video in real time.

Audio is bi-directional; sending "talk-back" audio back to the server functionality is available, all you need is a microphone.





Tutorial

Overview

In this section, we will show some simplistic approaches in deploying the ProximaWeb ActiveX control, LinearisWeb ActiveX control and SpectivaWeb ActiveX control. As ActiveX controls can be conveniently used in many different development environment, we will provide necessary information to use these components in the most common IDEs.

A quick tour of more powerful functionalities will be presented.

Hands on – Borland Delphi and C++ Builder

Importing the ActiveX Control

You can import the “Spectiva ActiveX Control” or “Linearis ActiveX Control” or “Proxima ActiveX Control” in Delphi as described in the Delphi documentation, which is partially reported here:

1. Choose “*Component → Import ActiveX Control...*” menu item.
2. Select the “*Spectiva Web Control Module*” type library from the list or “*Cieffe Proxima ActiveX Control module*”. The dialog lists all the registered libraries that define ActiveX controls (this is a subset of the libraries listed in the Import Type Library dialog). If the type library is not in the list, choose the Add button, find and select the SpectivaWeb.ocx or LinearisWeb.ocx or ProximaWeb.ocx control file, choose OK. This registers the type library, making it available. Then repeat step 2. You can then select and then click the “*Install*” button.
3. If you do not want to install the ActiveX control on the Component palette, choose Create Unit. This generates the TypeLibName_TLB unit and adds the declaration of its component wrapper. This exits the Import ActiveX dialog. In the following dialog, you can select the “*Into new package*” page tab and give a name, for example “*Insignis*”.
4. If you want to install the ActiveX control on the Component palette, select the Palette page on which this component will reside (usually: “*Insignis*”) and then choose Install. This generates the TypeLibName_TLB unit, like the Create Unit button, and then displays the Install component dialog, letting you specify the package where the components should reside (either an existing package or a new one).
5. When you exit the Import ActiveX dialog, the new TypeLibName_TLB unit appears in the directory specified by the Unit directory name box. This file contains declarations for the elements defined in the type library, as well as the generated component wrapper for the ActiveX control.
6. If you installed the generated component wrapper, an ActiveX control now resides on the Component palette. You can use the Object Inspector to set properties or write event handlers for this control.

Hands on – Javascript

Hosting the ActiveX

The first pass to handle a connection with a remote server is instantiate an ActiveX object in the html page. You can put the following code in the body section of your page.

```
<OBJECT ID="VisionWeb1" WIDTH=345 HEIGHT=282  
CLASSID="CLSID:10458b03-35ac-4d5c-b9aa-9645f27b3e4d"  
codebase="ProximaVisionWEB.cab#version=3,0,0,27">
```

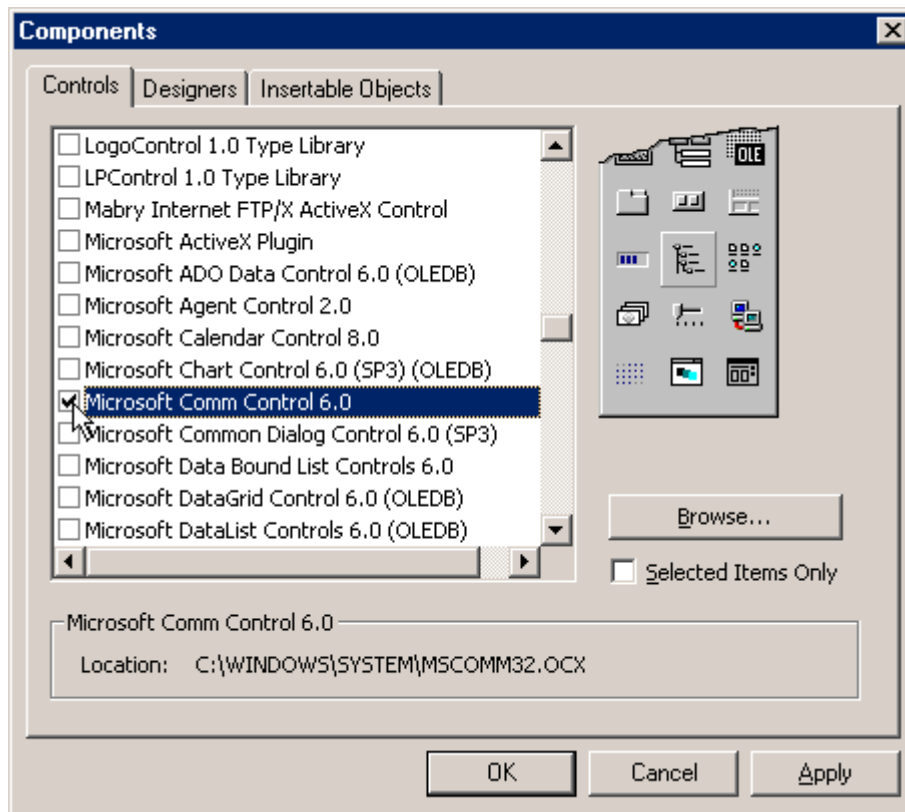
Hands on – Microsoft Visual Basic 6

Importing the ActiveX Control

You can import the “Spectiva ActiveX Control” or “Linearis ActiveX Control” or “Proxima ActiveX Control” in Visual Basic as described in the MSDN documentation, which is partially reported here:

1. On the **Project** menu, click **Components** to display the **Components** dialog box, as seen in figure 1.
2. Items listed in this dialog box include all registered insertable objects, designers, and ActiveX controls.
3. To add an ActiveX control to the Toolbox, select the check box to the left of the control name.
4. Click **OK** to close the **Components** dialog box. All of the ActiveX controls that you selected will now appear in the Toolbox.

Figure 1. The Components dialog box



To add ActiveX controls to the Components dialog box, click the Browse button, and locate files with name ProximaWeb.ocx and/or SpectivaWeb.ocx file name. These files are commonly installed in your \Windows\System or \Windows\System32 directory. When you add an ActiveX control to the list of available controls, Visual Basic automatically selects its check box in the Components dialog box.

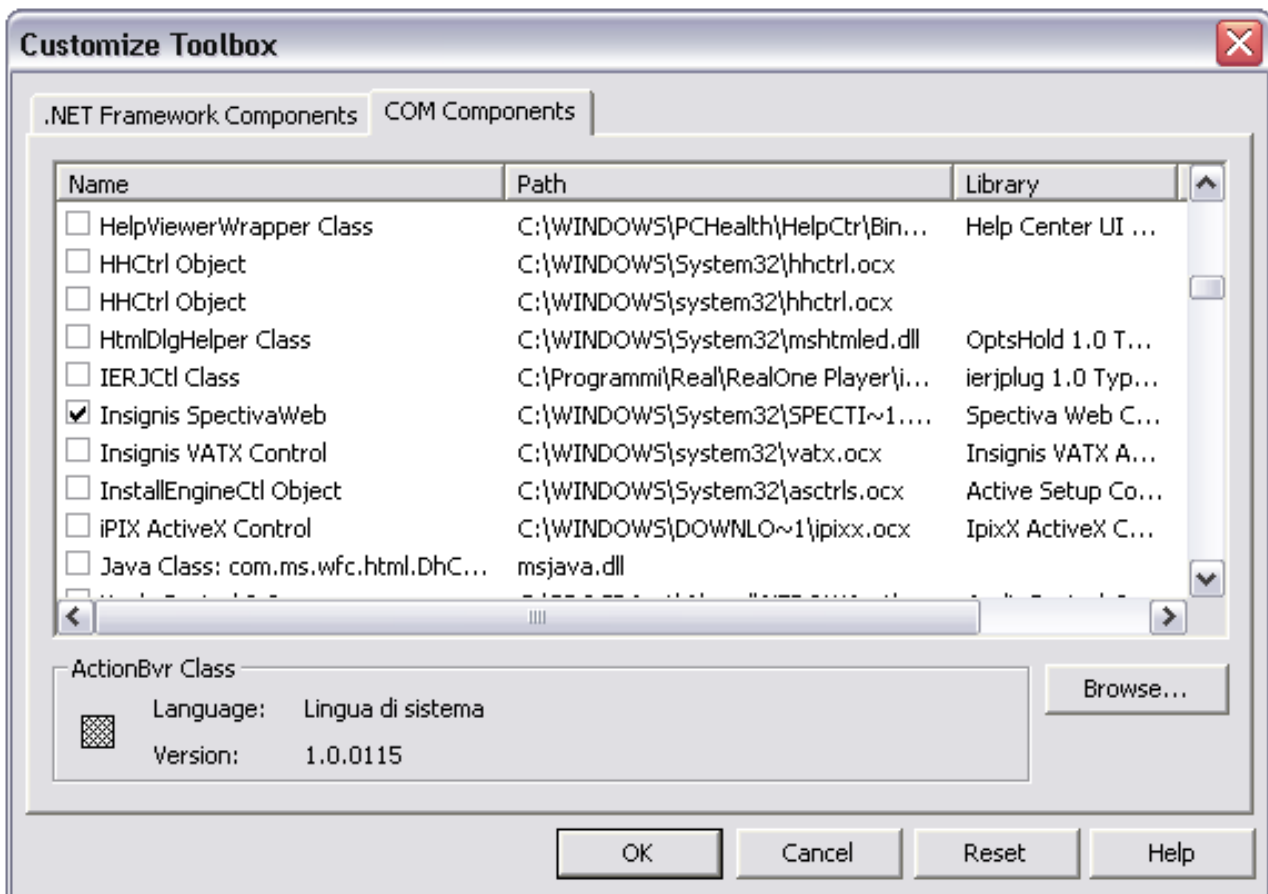
Hands on – Microsoft Visual C++ environment

Importing the ActiveX Control

You can import the Spectiva ActiveX control or Linearis ActiveX control or Proxima ActiveX control in Visual C++ as described in the MSDN documentation, which is partially reported here:

1. Create your new Visual C++ MFC application.
2. Right Click on the ToolBox; as the pop-up menu appears choose “Add/Remove items...”.
3. Select “COM Components” tab.
4. Select “Insignis Spectiva Web” or “Cieffe Linearis” or “Cieffe VisionWeb” ActiveX controls, As seen in fig. 2.

Figure 2: Customize Toolbox Dialog Window.



If this controls are not present in the COM Components list, just click “Browse...” button and select “SpectivaWeb.ocx” or “LinearisWeb.ocx” or “ProximaWeb.ocx”.

5. A new graphic control will be inserted in your ToolBox



Reference

Overview

This section contains a complete reference of Proxima ActiveX control, Linearis ActiveX control and Spectiva ActiveX control, grouped in the following topics:

- METHODS
- EVENTS

Each topic describes the features provided by the ActiveX control, listed in alphabetical order.

Items defined as **DEPRECATED** are not supported and could be removed in future versions, so they **MUST NOT BE USED** in any application.

Items defined as **UNUSED** or **INTERNAL ONLY** are reserved for internal use by Insignis Technologies and are not publicly documented.

METHODS

The following methods are provided by the dispatch interface of the Proxima and Spectiva ActiveX controls. Methods are here listed in alphabetical order, but in the full reference they will be presented grouped by logical function.

Function groups:

- Connection
- Windows Management
- Playback
- AUX and Alarms management
- Audio
- Dome Piloting
- Spot monitors
- Export
- Text Insertion
- Manual Recording
- Panorama Display

Method	Description	Group
AddToMPList	Adds the specified window to the multiple playback window list.	Playback
AddWinToPanorama		Panorama
BringWindowToFront	*** Proxima only, not needed for Spectiva	
CanSetPriorityCamera	Asks the server if a priority camera can be set	Connection
CaptureWindowEvents	Switches between automatic or manual dome control	Dome
ChangeAdaptiveQuality	Sets the appropriate image quality for adaptive transmission	Connection
ChangeAdaptiveRes	Sets the appropriate image resolution for adaptive transmission	Connection
ChangeCamera	DEPRECATED	
ChangeCameraWin	DEPRECATED	
ChangeDeltaThreshold	Set the current quality for delta or adaptive transmission on all the windows	Connection
ChangeDeltaThresholdWin	Sets the current quality for delta or adaptive transmission on the selected window	Connection
ChangeHwndWin	Changes the real Windows HWND of a specified window.	Window
ChangeResolutionWin	Changes the resolution of the given window.	Window
ChangeSector	DEPRECATED	
ChangeSectorWin	Changes sector to playback in the selected window.	Playback
ClearMPList	Removes all windows from multiple playback windows list.	Playback

Method	Description	Group
ClearWindow	*** Proxima only, not needed for Spectiva	window
Close	DEPRECATED	
Connect	Establishes a connection to the given host.	Connection
ConnectEx	Establishes a connection to the given host using the specified port.	Connection
ConnectUsingProxy	Establishes a connection to the given host using the specified port and a proxy.	Connection
ConvertImageToBitmap	It gives a bitmap containing the last image received and relative infos.	Playback
ConvertImageToBitmapWin	It gives a bitmap containing the last image received for the selected camera and connection and relative infos.	Playback
ConvertImageToBitmapWinEx	It gives a bitmap containing the last image received for the selected camera and connection and relative infos, specifying the desired resolution.	Playback
CreateGrid	Creates a grid of n columns and m rows in the display window.	Window
CreateWin	Creates a window in a grid.	Window
CryptConnect	Establish a connection to the given host Using encryption for user name and password.	Connection
DeltaTransmission	Turns ON or OFF the delta transmission for the given connection.	Connection
DestroyExternalWin	Removes the given window from the window list.	Window
DestroyPanoramaWin		Panorama
Disconnect	Closes selected connection.	Connection
EnableAudio	Enables or disables audio for a window.	Window
EnableDomeControlWin	Enables the dome PTZ control on the given window.	Dome
EnableMpegD1QualityEnhanceWin	Enhances motion details.	Playback
VideoQualityEnhanceWin	Improves image quality.	Playback
ExportImage	DEPRECATED	
ExportPause	Pauses the currently running export process.	Export
ExportResume	Resumes a paused export process.	Export
ExportStart	Sets the starting point of a video footage to export.	Export
ExportStop	Sets the ending point of a video footage to export.	Export
FocusDome	Sets focus far or near.	Dome
GetAlarmName	Returns the name of an alarm.	AUX/Alarms
GetCodecType	Returns the encoding algorithm of the given window.	Connection
GetExportInfo	INTERNAL USE	Export
GetFindData	Asks the server the data necessary to fill the search bar(s).	Playback

Method	Description	Group
GetHeightResolution	Returns the resolution height of the camera displayed in the given window.	Window
GetImageInfo	DEPRECATED	
GetMPList	Returns a list of windows actually set in multiple playback windows list.	Playback
GetMPMaster	Returns the window index of the window set as master in multiple playback.	Playback
GetRecordingMode	Asks the recording state for the specified camera and sector.	Manual recording
GetVideoStandard	Returns the video standard of the server.	Connection
GetTextInsertionOptions	Gets current display options for text insertion.	Text Insertion
GetTextInsertionOptionsEx	Gets current extended display options for text insertion.	Text Insertion
GetWidthResolution	Returns the resolution Width of the camera displayed in the given window.	Connection
GetWinFromPoint		Panorama
KeepAspectRatioWin	Forces the selected window to keep the original aspect ratio.	Window
Init	DEPRECATED	
InvertRegion	INTERNAL USE	
IrisDome	Opens or closes the dome iris.	Dome
MoveDome	Moves the dome with given horizontal and vertical speed.	Dome
Play	Starts playback in the selected window.	Playback
PlayBegin	Jumps to the beginning of the recorded material of the actual camera.	Playback
PlayEnd	Jumps to the end of the recorded material of the actual camera.	Playback
PlayFfw	Starts playback fast forward.	Playback
PlayFind	Starts playback from the selected time (time_t).	Playback
PlayFind64	Starts playback from the selected time expressed in _int64.	Playback
PlayFindT	Starts playback from the selected time, specified in single time components.	Playback
PlayLocal	Starts playback of the given local video file.	Playback
PlayRew	Starts playback rewind.	Playback
PlayStill	Pauses the playback.	Playback
ReceiveStatusAuxAlarm	Asks the actual status of AUX and alarms.	AUX/Alarms
RemoveFromMPList	Removes the specified window from the multiple playback window list.	Playback
RemoveWinFromPanorama		Panorama

Method	Description	Group
RequestStreamInfo	Asks to periodically send (or stop sending) infos about the video stream.	Connection
ResetManualRec	Switches to Custom Scheduler.	Manual recording
ResetCustomDataSP	Resets all custom data for the selected spot monitor.	Spot
ResetSpotMonitor	DEPRECATED	
ResizeWindow	Sets a new size to the selected window.	Window
SaveBitmapWinToFile	Writes the currently displayed frame exactly as shown on screen, automatically opening a "save file" dialog.	Playback
SaveBitmapWinToFileEx	Writes the currently displayed frame exactly as shown on screen with custom image size, automatically opening a "save file" dialog.	
SelectPresetDome	Moves the dome to a selected preset.	Dome
SelectTourDome	Starts a dome tour.	Dome
SendAlarm	Sets the status of the specified alarm.	AUX/Alarms
SendAux	Sets the status of the specified AUX.	AUX/Alarms
SendCameraAux	Sets the status of the specified Camera AUX.	AUX/Alarms
SendCertificate	Returns a decrypted buffer to the server.	Connection
SendCertificateStatus		Connection
SendCertificateBuffer		Connection
SendDeviceTextInsertionData	Sends a string to be inserted in the storage.	Text Insertion
SendImages	Asks the server to start or stop the images visualization.	Connection
SendPacket	INTERNAL USE	Connection
SendSerialPacket	INTERNAL USE	Connection
SetAppType	INTERNAL USE	
SetAudioVolume	Sets the audio volume of the given channel of the given connection.	Audio
SetBidirectionalAudioVolume	Sets the volume of the talk channel from the client to server.	Audio
SetCamera	Sets the camera to display for the selected window of the selected connection.	Window
SetConnectionPriority	Allows to increase or decrease the priority of the visualization thread.	Connection
SetCustomCameraSP	Sets the selected camera in the selected spot monitor cycle, specifying whether to show or hide overlay text.	Spot
SetCustomDataSP	Fine specify settings for spot monitor cycle.	Spot
SetExportType	INTERNAL USE	
SetExternalLogo	DEPRECATED	
SetExternalWin	Adds the given window from the window list.	Window
SetFontScaleWin	Sets the font scale of the overlay texts.	Window

Method	Description	Group
SetFullscreenWin	INTERNAL USE	
SetMainWindow	DEPRECATED	
SetMPMaster	Sets selected window as Master window in multiple playback.	Playback
SetPanoramaWin		Panorama
SetPriorityCamera	Sets a priority camera.	Connection
SetSmartCardStatus		Connection
SetSpotMonitor	DEPRECATED	
SetTalkVolume	Changes the volume of the talk audio channel,	Audio
SetTextInsertionOptions	Sets necessary options to display text insertion data.	Text Insertion
SetTextInsertionOptionsEx	Sets necessary options to display text insertion data, with more options available then SetTextInsertionOptions	Text Insertion
SetVisualizationStatus	DEPRECATED	
SetWindowText	DEPRECATED	
SetWindowTextSize	DEPRECATED	
SetWinPanoramaOptions		Panorama
SetZoom	Sets the virtual zoom on the current window.	Window
ShowBandwidthWin	Shows/Hides bandwidth use in the overlay text.	Window
ShowMetadataWin	Shows/Hides visual data regarding the behavioral analysis.	Window
StartCustomSP	Starts the custom spot monitor cycle.	Spot
StartManualRec	Forces the recording to start for the specified connection, camera and sector.	Manual recording
Stop	Stops playback in the selected window.	Playback
StopManualRec	Switch back to the standard recording scheduler.	Manual recording
StopCustomSP	Stops the custom spot monitor cycle.	Spot
WriteText	DEPRECATED	
ZoomDome	Zoom far or near the selected dome.	Dome

CONNECTION Methods

This group contains all the methods necessary to establish, manage and break a connection to a DVMS video server.

- Connect
- ConnectEx
- ConnectUsingProxy
- CryptConnect
- Disconnect
- DeltaTransmission
- ChangeAdaptiveQuality
- ChangeAdaptiveRes
- ChangeDeltaThresholdWin
- GetVideoStandard
- RequestStreamInfo
- SendImages
- SendCertificate
- SendCertificateStatus
- SendCertificateBuffer
- SetSmartCardStatus
- SetConnectionPriority
- CanSetPriorityCamera
- SetPriorityCamera
- GetCodecType

Connect Method

Purpose

Connect to a DVR server specifying its IP address, delta type, user name and password.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

```
long Connect(BSTR _user, BSTR _passwd, BSTR _host, long _delta, short _keyFrameRefresh, short _bitRate);
```

Input parameters

Type	Name	Description
BSTR	_user	name of user for server login.
BSTR	_passwd	user password.
BSTR	_host	IP address of DVMS server to connect to.
long	_delta	Boolean set to 0 to deactivate delta transmission activate otherwise.
short	_keyFrameRefresh	Used only by ProXima2 Servers using delta; it sets the max distance between 2 keyframes
short	_bitRate	Used only by ProXima2 Servers using delta; it asks for an average bitrate to be kept

Return

unique ID for the server connection.

-1 if a generic error occurred during connection, or max supported connection limit already reached.

Remarks

Using this method to connect to a Spectiva server encryption is automatically used, while to use password encryption connecting to a Proxima or Linearis server it is necessary to use CryptConnect method.

See also

ConnectEx

ConnectUsingProxy

Disconnect

CryptConnect



ConnectEx Method

Purpose

Connect to a Spectiva DVR server specifying its IP address, delta type, port number, user name and password.

Compatibility

Proxima: no
Linearis: no
Spectiva: yes

Syntax

```
long ConnectEx(BSTR _user, BSTR _passwd, BSTR _host, long _delta, short  
_keyFrameRefresh, short _bitRate, long _port );
```

Input parameters

Type	Name	Description
BSTR	_user	name of user for server login.
BSTR	_passwd	user password.
BSTR	_host	IP address of DVMS server to connect to.
long	_delta	Boolean set to 0 to deactivate delta transmission activate otherwise.
short	_keyFrameRefresh	Used only by ProXima2 Servers using delta; it sets the max distance between 2 keyframes
short	_bitRate	Used only by ProXima2 Servers using delta; it asks for an average bitrate to be kept
long	_port	The number of the port to connect.

Return

unique ID for the server connection.
-1 if a generic error occurred during connection, or max supported connection limit already reached.

Remarks

Using this method to connect to a Spectiva server encryption is automatically used.

See also

Connect
ConnectUsingProxy
Disconnect



CryptConnect



ConnectUsingProxy Method

Purpose

Connect to a DVR server specifying its IP address, delta type, port number, user name and password, and a proxy server.

Compatibility

Proxima: no
Linearis: yes
Spectiva: yes

Syntax

long ConnectUsingProxy(BSTR *user*, BSTR *pass*, BSTR *host*, long *port*, short *proxyType*, BSTR *proxyAddr*, long *proxyPort*, BSTR *proxyUser*, BSTR *proxyPass*, long *delta*, short *keyFrameRefresh*, short *bitRate*);

Input parameters

Type	Name	Description
BSTR	user	name of user for server login.
BSTR	pass	user password.
BSTR	host	IP address of DVMS server to connect to.
long	port	The number of the port to connect.
short	proxyType	Type of proxy: <ul style="list-style-type: none">● 0 = no proxy● 1 = socks4 proxy● 2 = socks4a proxy● 3 = socks5 proxy
BSTR	proxyAddr	Proxy address
long	proxyPort	Proxy port
BSTR	proxyUser	Proxy user name (can be empty)
BSTR	proxyPass	Proxy password (can be empty)
long	delta	Boolean set to 0 to deactivate delta transmission activate otherwise.
short	keyFrameRefresh	Used only by ProXima2 Servers using delta; it sets the max distance between 2 keyframes
short	bitRate	Used only by ProXima2 Servers using delta; it asks for an average bitrate to be kept

Return

unique ID for the server connection.



-1 if a generic error occurred during connection, -2 if max supported connection limit already reached, -3 if out of memory or system resources.

Remarks

Using this method to connect to a Spectiva server encryption is automatically used.

See also

Connect
ConnectEx
Disconnect
CryptConnect



CryptConnect Method

Purpose

Connect to a DVR server specifying its IP address, delta type, user name and password, using a strong encryption model.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: no

Syntax

```
long Connect(BSTR _user, BSTR _passwd, BSTR _host, long _delta, short _keyFrameRefresh, short _bitRate);
```

Input parameters

Type	Name	Description
BSTR	_user	name of user for server login.
BSTR	_passwd	user password.
BSTR	_host	IP address of DVMS server to connect to.
long	_delta	Boolean set to 0 to deactivate delta transmission activate otherwise.
short	_keyFrameRefresh	Used only by ProXima2 Servers using delta; it sets the max distance between 2 keyframes
short	_bitRate	Used only by ProXima2 Servers using delta; it asks for an average bitrate to be kept

Return

unique ID for the server connection.

-1 if a generic error occurred during connection.

-2 if the specified server type is not valid.

See also

Disconnect

Connect

ConnectEx

ConnectUsingProxy

Remarks

This function is only supported by Proxima3 release 3.00 or later. To connect to a Proxima 2 video server or to a Proxima 3 beta or RC, use the Connect method.



Disconnect Method

Purpose

Close the connection to the specified DVR server.

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

long Disconnect (long _position)

Input parameters

Type	Name	Description
long	_position	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() methods.

Return

-1 if the given connection ID is not valid or it's a local video footage playback connection.
1 on success.

See also

Connect
ConnectEx
ConnectUsingProxy
CryptConnect

DeltaTransmission Method

Purpose

Asks the DVR Server to activate or deactivate the differential video transmission for the selected connection.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

```
long DeltaTransmission(long _activate, long _position)
```

Input parameters

Type	Name	Description
long	_activate	Boolean: 0 for normal wavelet transmission 1 for delta-wavelet
long	_position	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() methods.

Returns

1 on success.

-1 if the given connection ID is not valid

Remarks

The differential transmission algorithm depends on the version of the software installed on the video server:

Proxima 2 uses DeltaWLT;

Proxima 3 RC1 to 3.00.02 uses DeltaHF;

Proxima 3 release 3.00.03 and above uses DeltaV4 (Enpacta);

Spectiva uses DeltaHF;

Spectiva 2 uses DeltaV4 (Enpacta);

ChangeAdaptiveQuality Method

Purpose

Call this method to set the appropriate image quality (compression) to use with the adaptive transmission.

Compatibility

Proxima: no
Linearis: yes
Spectiva: yes

Syntax

long ChangeAdaptiveQuality (long _connection, long _window, long _quality)

Input parameters

Type	Name	Description
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() methods.
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long	_quality	Quality of the compression to apply. Valid Range: [0 .. 100]

Returns

1 on success.
-1 if the given connection ID or window is not valid

ChangeAdaptiveRes Method

Purpose

Call this method to set the appropriate image resolution to use with the adaptive transmission.

Compatibility

Proxima: no
Linearis: yes
Spectiva: yes

Syntax

long ChangeAdaptiveRes (long _connection, long _window, long _resolution)

Input parameters

Type	Name	Description
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() methods.
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long	_resolution	Resolution of the adaptive stream to send Valid values are: 0: Currently configured resolution on server 1: D1 resolution 2: HD1 resolution 3: CIF resolution 4: QCIF resolution

Returns

1 on success.
-1 if the given connection ID or window is not valid

ChangeDeltaThresholdWin Method

Purpose

Call this method to set the appropriate image resolution to use with the adaptive transmission.

Compatibility

Proxima: no
Linearis: no
Spectiva: yes

Syntax

```
long ChangeDeltaThresholdWin (long _thresholdValue, long _connection, long _window)
```

Input parameters

Type	Name	Description
long	_thresholdValue	Threshold value: 1: Low quality 2: Medium quality 3: High quality [4..100] Custom quality value.
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() methods.
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.

Returns

1 on success.
-1 if the given connection ID or window is not valid

GetVideoStandard Method

Purpose

Asks the video standard in use on the DVR Server.

Compatibility

Proxima: no

Linearis: yes

Spectiva: yes

Syntax

long GetVideoStandard(long connection)

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() methods.

Returns

0 if the video standard is PAL.

1 if the video standard is NTSC.

-1 if the given connection ID is not valid

RequestStreamInfo Method

Purpose

Asks the DVR server to periodically send to the client basic stream informations about the video shown in the specified window.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

```
long RequestStreamInfo(long window, long* date, long interval);
```

Input parameters

Type	Name	Description
long	window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long*	date	Pointer to an __int64 containing the date of the frame currently visualized in the given window.
long	interval	Period, expressed in milliseconds, of the OnStreamInfo event used to return the requested informations; use 0 to stop.

Return

1 on success

0 if "window" parameter points to an invalid or not existing window.

Remarks

If "interval" parameter is greater then 0 the video server tries to post the required infos to the client every "interval" milliseconds, but this period could increase in case of heavy workload.

When these infos are received an OnStreamInfo event is triggered.

See also

OnStreamInfo Event.

SetExternalWin method.



SendImages Method

Purpose

Orders to the DVR Server to Start or Stop sending video streams to the client connected to the specified connection.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

```
long SendImages(long _connection, long _status)
```

Input parameters

Type	Name	Description
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() methods.
long	_status	Boolean set to 1 to start image transmission, 0 to stop it.

Return

1 on success.

-1 if the given connection ID is not valid or if it's an ID of a refused connection.

SendCertificate Method

Purpose

This has to be called as answer method to the OnNeedCertificate event to send back the decrypted buffer.

Compatibility

Proxima: no
Linearis: yes
Spectiva: yes

Syntax

long SendCertificate(long * buffer, long size, long connection)

Input parameters

Type	Name	Description
long*	buffer	A buffer containing decrypted data to be sent back to the server as user validation
long	size	Size in bytes of the data structure pointed at *buffer.
long	connection	ID of server connection.

Return

1 on success.
-1 if the given connection ID is not valid or if it's an ID of a refused connection.

SendCertificateStatus Method

Purpose

This method has to be called to notify to the server if the specified certificate is present within the reader.

Compatibility

Proxima: no
Linearis: no
Spectiva: yes

Syntax

```
long SendCertificateStatus(long* serial, long serialSize, long status);
```

Input parameters

Type	Name	Description
long*	serial	A data buffer containing the certificate serial.
long	serialSize	Size in bytes of the certificate serial data buffer.
long	status	Boolean set to 1 if the certificate is present or 0 if it is NOT present,

Return

1 on success.
-1 if the given input data is erroneous (serial is NULL or serialSize is 0)

SendCertificateBuffer Method

Purpose

This has to be called as answer method to the OnNeedCertificateBuffer event to send back the decrypted buffer.

Compatibility

Proxima: no
Linearis: no
Spectiva: yes

Syntax

long SendCertificateBuffer(long* serial, long serialSize, long* buffer, long size, long connection)

Input parameters

Type	Name	Description
long*	serial	A data buffer containing the certificate serial.
long	serialSize	Size in bytes of the certificate serial data buffer.
long*	buffer	A buffer containing decrypted data to be sent back to the server as user validation.
ling	size	Size in bytes of the data structure pointed at *buffer.
long	status	Boolean set to 1 if the certificate is present or 0 if it is NOT present,

Return

1 on success.
-1 if the given input data is erroneous (serial or buffer is NULL or serialSize or size is 0)

SetSmartCardStatus Method

Purpose

Use this method to notify if the smart card is present within the card reader.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: no

Syntax

```
long SetSmartCardStatus(long status)
```

Input parameters

Type	Name	Description
long	status	Boolean set to 1 if the certificate is present or 1 if it is NOT present,

Return

1 on success.

0 on error,

SetConnectionPriority Method

Purpose

Set the priority of the thread that manages the connection with the server. As this thread owns, among the other things, the messages dispatcher, increasing this thread priority could increase ActiveX reactivity on incoming events.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

```
long SetConnectionPriority(long connection, long priority)
```

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() methods.
long	priority	Valid range: [0..2] 0 = above normal priority 1 = highest priority 2 = time critical priority

Return

1 on success.

0 if the given connection ID is not valid or if it's an ID of a refused connection.

Remarks

Suggested values are:

0 for slow machines (2 Ghz or behind)

1 for mid-range machines (2 – 3 Ghz)

2 for high range machines (3 Ghz and above) and only in application where real time events management is strictly required. It is not suggested to set the priority as time critical for more than one connection at the same time.

CanSetPriorityCamera Method

Purpose

Asks the server if a priority camera can be set.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

```
long CanSetPriorityCamera(long connection)
```

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() methods.

Return

1 if the server is multiplexed and a priority camera can be set.

0 if the server is not multiplexed.

-1 if the given connection ID is not valid or if it's an ID of a refused connection.

-2 if the user doesn't have permission to change the priority of cameras.

See also

SetPriorityCamera method.

OnSetPriorityCamera event.

SetPriorityCamera Method

Purpose

Set the priority of the selected camera on multiplexed servers.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

```
long SetPriorityCamera(long connection, long camera)
```

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() methods.
long	camera	The number of the camera to set the priority to.

Return

1 on success.

-1 if the given connection ID is not valid or if it's an ID of a refused connection or if the camera is not valid.

Remarks

Use -1 for the camera parameter to reset the priority.

See also

CanSetPriorityCamera method.

OnSetPriorityCamera event.

GetCodecType Method

Purpose

Returns the encoding algorithm of the specified camera

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

long GetCodecType(long connection, long camera)

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() methods.
long	camera	The number of the camera.

Return values:

0: Wavelet
1: Enpacta
2: MPEG4
3: H.264
4: JPEG
5: JPEG 2000

WINDOWS MANAGEMENT Methods

This group contains all the methods necessary to manage windows size and resolution, to map Window handles to internal references, and to prepare grid useful for quad and custom display.

- CreateGrid
- CreateWin
- SetCamera
- ChangeResolutionWin
- GetHeightResolution
- GetWidthResolution
- SetFontScaleWin
- SetZoom
- ResizeWindow
- KeepAspectRatio
- SetExternalWin
- DestroyExternalWin
- ChangeHwndWin
- EnableAudio
- ShowBandwidthWin
- ShowMetadataWin

CreateGrid Method

Purpose

Splits the ActiveX client area into a grid of `_ncol` columns and `_nrow` rows. This allow many different video streams to be displayed in the same window.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

```
long CreateGrid(short _ncol, short _nrow);
```

Input parameters

Type	Name	Description
short	<code>_ncol</code>	Number of columns. Valid range [0..12].
short	<code>_nrow</code>	Number of rows. Valid range [0..12].

Return

1 on success

Remarks

If `_ncol` or `_nrow` is out of the valid range, it is cut to the nearest valid value. Once the grid is created it is still necessary to create visualization windows within the grid to display video streams. To create these windows within this grid you can use the `CreateWin` method.

See also

`CreateWin()` method.

CreateWin Method

Purpose

Create a display window within a grid of ActiveX client area. This grid must be created by CreateGrid method.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

```
long CreateWin(short _x1, short _y1, short _x2, short _y2)
```

Input parameters

Type	Name	Description
short	_x1	Abcissas coordinates of the upper left corner of the client window
short	_y1	Ordinates coordinates of the upper left corner of the client window
short	_x2	Abcissas coordinates of the lower right corner of the client window
short	_y2	Ordinates coordinates of the lower right corner of the client window

Return

unique ID of the window created

-1 if the grid was not initialized or if the MAX_WINDOWS limit has been reached.

Remarks

Before creating a window using CreateWin it is necessary to initialize the grid using the CreateGrid method.

Abcissas and Ordinates coordinates are NOT pixel references, but numbers of

columns and rows in the grid.

See also

CreateGrid

Example

Creation of a grid and grid setup for quad display.

```
VisionWeb1->CreateGrid(12,12);  
win4_1 = VisionWeb1->CreateWin(0,0,6,6);  
win4_2 = VisionWeb1->CreateWin(6,0,12,6);  
win4_3 = VisionWeb1->CreateWin(0,6,6,12);  
win4_4 = VisionWeb1->CreateWin(6,6,12,12);
```

SetCamera Method

Purpose

Displays video streams from a camera of the specified connection in the window passed as parameter, or remove the camera from the window.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

SetCamera(short _numTele, short _connection, long _window)

Parameters

Type	Name	Description
short	_numTele	ID of camera
short	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.

Return

1 on success.

-1 if a generic error occurred.

Remarks

Use -1 as _numTele parameter to remove the camera display from the specified window.

ChangeResolutionWin Method

Purpose

Asks the DVR server to switch video resolution of the video displayed in the given window between QCIF, CIF, 3/4 FULL and FULL resolution.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

```
long ChangeResolutionWin(long _resolution, long _win);
```

Input parameters

Type	Name	Description
long	_resolution	Constant indicating the chosen resolution. Valid range [0..3] See Remarks for constant to pixels resolution map.
long	_win	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.

Return

1 on success
-1 if a generic error occurred.

Remarks

The following table shows _resolution values and image resolutions.

_resolution	Width	NTSC Height	PAL Height
NORMAL_RESOLUTION = 0	360	243	288
CIF_RESOLUTION = 1	176	122	144
QCIF_RESOLUTION = 2	90	62	72
MAX_RESOLUTION = 3	720	243	288

Delta connections do NOT support MAX_RESOLUTION.

If ChangeResolutionWin is called with MAX_RESOLUTION for a window belonging to a connection using differential transmission, it is used

NORMAL_RESOLUTION instead.

If DVR server is using DeltaWavelet, no more than ONE window for each connection can use NORMAL_RESOLUTION; all the other windows will use CIF_RESOLUTION. This limit does not exist if the DVR server is using Enpacta as differential transmission protocol.

This function has no effect on cameras recorded in Enpacta or Mpeg4 when the DVR server is using hardware compression.



GetHeightResolution Method

Purpose

Returns the height of the specified window.

Compatibility

Proxima: yes (since rel 4.03)
Linearis: yes
Spectiva: yes

Syntax

```
long GetHeightResolution(long _connection, long _win);
```

Input parameters

Type	Name	Description
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	_win	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.

Return

Height (in pixels) of the current window .

GetWidthResolution Method

Purpose

Returns the width of the specified window..

Compatibility

Proxima: yes (since rel 4.03)

Linearis: yes

Spectiva: yes

Syntax

```
long GetWidthResolution(long _connection, long _win);
```

Input parameters

Type	Name	Description
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	_win	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.

Return

Width (in pixels) of the current window.

SetFontScaleWin Method

Purpose

Call this method to set the appropriate font size of the overlay text

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

ResizeWindow(long _window, long _scale)

Input parameters

Type	Name	Description
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long	_scale	Scale factor referred to the base font size. 1: normal font size 2: double font size 3: triple font size n: n times the base font size

Return

1 on success
-1 if a generic error occurred.

SetZoom Method

Purpose

Use this method to realize a digital zoom effect on live and playback video. The images are cropped at the specified coordinates, then the resulting image is stretched to fill the window

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

```
SetZoom(long _window, long _status, long _left, long _top, long _right, long  
_bottom)
```

Input parameters

Type	Name	Description
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long	_status	Boolean set to 1 to enable this feature, 0 to turn it off.
long	_left	Left border of the crop rectangle (use percentage referred to window width)
long	_top	Top border of the crop rectangle (use percentage referred to window height)
long	_right	Right border of the crop rectangle (use percentage referred to window width)
long	_bottom	Bottom border of the crop rectangle (use percentage referred to window height)

Return

1 on success
-1 if a generic error occurred.

ResizeWindow Method

Purpose

Changes the size of the window rectangle used to display video streams.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: no (automatically detected)

Syntax

ResizeWindow(long _window, long _x, long _y, long _width, long _height)

Input parameters

Type	Name	Description
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long	_x	No more used; suggested value:0.
long	_y	No more used; suggested value:0.
long	_width	Pixel width of the display rectangle
long	_height	Pixel height of the display rectangle

Return

0 on success

Remarks

These parameters represent the real width and height used for the video rendering. They're not necessary connected to the video resolution.

A low resolution video stream could be displayed at 800x600 if the following methods are called:

```
#define QCIF_RESOLUTION 2  
VisionWeb1->ChangeResolutionWin (QCIF_RESOLUTION, windowID);  
VisionWeb1->ResizeWindow (windowID, 0, 0, 800, 600);
```

Nevertheless it is suggested to keep video resolution and window size as congruent as possible.

KeepAspectRatio Method

Purpose

Forces the video to keep the native aspect ratio. If the drawing area of the window does not respect the ratio, black filling bars will be inserted.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

```
long KeepAspectRatio (long _status);
```

Input parameters

Type	Name	Description
long	_status	Boolean set to 1 to enable this feature, 0 to turn it off.

Return

1 on success

-1 if a generic error occurred.

SetExternalWin Method

Purpose

Set a generic window (not belonging to the ActiveX) as a display window to use the show a video stream of a specified connection.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

```
long SetExternalWin(long _handle, long _position);
```

Input parameters

Type	Name	Description
long	_handle	Real window HWND
long	_position	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Return

unique ID for the window.
-1 if a generic error occurred.

Remarks

The _position parameter is unused by the Spectiva ActiveX, so it can be always set to 0 or any other value.

DestroyExternalWin Method

Purpose

Remove the specifies window from the list of windows used by the ActiveX.

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

```
long DestroyExternalWin (long _handle);
```

Input parameters

Type	Name	Description
long	_handle	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.

Return

1 on success.
-1 if the specified windows does NOT exists in the given connection.
-2 if the specified windows does exist but it is NOT an external window.

Remarks

The _handle parameter is NOT the real windows handle, but the internal window ID.

ChangeHwndWin Method

Purpose

Replace the real window HWND associated with an internal window ID with an HWND belonging to another window.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

```
long ChangeHwndWin (long _handle, _long window);
```

Input parameters

Type	Name	Description
long	_handle	Target window handle
long	_window	Internal ID of the window whose HWND will be changed

Return

1 on success.

-1 if the specified windows does NOT exists in the given connection.

Remarks

This allow to move a camera display from a window to another without the need to create a new window or set another existing window as an ActiveX external window, set the camera... but just go on simply visualizing the A/V stream in another window.

ShowBandwidthWin Method

Purpose

Call this method to show or hide the bandwidth usage in the overlay text

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

```
long ShowBandwidthWin (long _window, long _status);
```

Input parameters

Type	Name	Description
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long	_status	1 to show the bandwidth usage 0 to hide the bandwidth usage

Return

1 on success.

0 if the specified windows does NOT exists.

ShowMetadataWin Method

Purpose

Call this method to show or hide the graphical results of the behavioral analysis

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

```
long ShowMetadataWin (long _window, long _what);
```

Input parameters

Type	Name	Description
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long	_what	This is a bit field value. Set this parameter to zero to hide the metadata display, otherwise compose a logical OR of the following constants to display the preferred analysis results: const SHOW_METADATA_TRACKING = 1; const SHOW_METADATA_PERMANENCY = 2; const SHOW_METADATA_MOTION = 4; const SHOW_METADATA_CHANGE = 8;

Return

1 on success.

0 if the specified windows does NOT exists, or the _what parameter is invalid.

Remarks

You also need to set these properties of the ActiveX control:

permanencyColor

motionColor

changeColor

The type of this properties is OLE_COLOR.

EnableAudio Method

Purpose

Enables or disables the audio for the selected window.

Compatibility

Proxima: no

Linearis: yes

Spectiva: yes

Syntax

```
long EnableAudio (long _window, long _status);
```

Input parameters

Type	Name	Description
long	_window	Internal ID of the window whose HWND will be changed
long	_status	1 to enable audio 0 to disable audio

Return

1 on success.

0 if the specified windows does NOT exists.

PLAYBACK Methods

This group contains all the methods necessary to playback recorded video footage, either in a single playback stream or, if supported by the DVMS server, multiple synchronized streams playback.

- Play
- Stop
- PlayBegin
- PlayEnd
- ChangeSectorWin
- PlayFind
- PlayFind64
- PlayFindT
- PlayStill
- PlayFw
- PlayRew
- PlayLocal
- AddToMPList
- RemoveFromMPList
- ClearMPList
- SetMPMaster
- GetMPMaster
- GetMPList
- GetFindData
- ConvertImageToBitmap
- ConvertImageToBitmapWinEx
- SaveBitmapWinToFile
- SaveBitmapWinToFileEx
- VideoQualityEnhanceWin
- EnableMpegD1QualityEnhanceWin

You can see also the following methods in CONNECTION group:

- SendImages
- DeltaTransmission
- RequestStreamInfo

Play Method

Purpose

Ask the DVR server to start sending recorded video for the camera actually visualized in the given window. It also start multiple playback, if the window belongs to a Spectiva server connection Multiple Playback has been properly prepared.

Compatibility

Proxima: yes (without Multiple Playback)

Linearis: yes (without Multiple Playback)

Spectiva: yes

Syntax

Play (long _window)

Input parameters

Type	Name	Description
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.

Return

1 on success.

-1 if a generic error occurred.

Remarks

See AddToMPList() method for VERY IMPORTANT remarks about multiple playback.

Stop Method

Purpose

Ask the DVR server to stop sending recorded video and switch back to live video.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

Stop (long _window)

Input parameters

Type	Name	Description
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.

Return

1 on success.

-1 if a generic error occurred.

PlayBegin Method

Purpose

Ask the DVR server to move the playback to the beginning of the recording of the active camera.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

PlayBegin (long _window)

Input parameters

Type	Name	Description
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.

Return

1 on success.

-1 if a generic error occurred.

Remarks

This method requires the window to be already in playback mode, so it's only possible to call the PlayBegin() method AFTER the Play() method has been called and no error was returned.

PlayEnd Method

Purpose

Ask the DVR server to move the playback to the end of the recording of the active camera.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

PlayEnd (long _window)

Input parameters

Type	Name	Description
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.

Return

1 on success.

-1 if a generic error occurred.

Remarks

This method requires the window to be already in playback mode, so it's only possible to call the PlayEnd() method AFTER the Play() method has been called and no error was returned.

ChangeSectorWin Method

Purpose

Change the sector to be displayed in the given window when in playback mode.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

```
long ChangeSectorWin(long _sector, long _window);
```

Input parameters

Type	Name	Description
long	_sector	Number of the sector to be played back.
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.

Return

1 on success.

-1 if a generic error occurred.

Remarks

Use following constants value for _sector:

Sector	Value
PRIME	0
TIME LAPS	1
ALARMS	2

PlayFind Method

Purpose

Ask the DVR server to move the playback at the given date and hour expressed using time_t date/time data type.

Compatibility

Proxima: yes

Linearis: no

Spectiva: yes

Syntax

```
long PlayFind(long _window, long _datetime)
```

Input parameters

Type	Name	Description
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long	_datetime	Date and Hour to playback. _datetime must be a valid time_t value.

Return

1 on success.

-1 if a generic error occurred.

Remarks

This method requires the window to be already in playback mode, so it's only possible to call the PlayFind() method AFTER the Play() method has been called and no error was returned.

If no recorded video is found at the given date and hour, it's played back the nearest video.

PlayFind64 Method

Purpose

Ask the DVR server to move the playback at the given date and hour expressed using `_int64` date/time data type..

Compatibility

Proxima: no
Linearis: yes
Spectiva: yes

Syntax

```
long PlayFind64(long _window, long _lowPart, long _highPart)
```

Input parameters

Type	Name	Description
long	<code>_window</code>	ID of a visualization window. This must be a valid window ID returned by <code>SetExternalWin()</code> or <code>CreateWin()</code> methods.
long	<code>_lowPart</code>	The 32 less significant bits of the <code>_int64</code> date/time
long	<code>_highPart</code>	The 32 most significant bits of the <code>_int64</code> date/time

Return

1 on success.
-1 if a generic error occurred.

Remarks

This method requires the window to be already in playback mode, so it's only possible to call the `PlayFind()` method AFTER the `Play()` method has been called and no error was returned.

If no recorded video is found at the given date and hour, it's played back the nearest video.

This method support random playback repositioning even using Multiple Playback.

PlayFindT Method

Purpose

Ask the DVR server to move the playback at the given date and hour, where each time component is expressed using single parameters.

Compatibility

Proxima: no
Linearis: yes
Spectiva: yes

Syntax

```
long PlayFindT(long _window, long _year, long _month, long _day, long _hour,  
long _minute, long _second)
```

Input parameters

Type	Name	Description
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long	_year	year
long	_month	Month [1..12]
long	_day	Day of the month [1..31]
long	_hour	Hour [0..23]
long	_minute	Minute [0..59]
long	_second	Seconds [0..59]

Return

1 on success.
-1 if a generic error occurred.

Remarks

This method requires the window to be already in playback mode, so it's only possible to call the PlayFindT() method AFTER the Play() method has been called and no error was returned.

If no recorded video is found at the given date and hour, it's played back the nearest video.

This method support random playback repositioning even using Multiple Playback.



PlayStill Method

Purpose

Ask the DVR server to pause the playback the record video.

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

PlayStill (long _window)

Input parameters

Type	Name	Description
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.

Return

1 on success.
-1 if a generic error occurred.

Remarks

This method requires the window to be already in playback mode, so it's only possible to call the PlayStill() method only AFTER the Play() method has been called and no error was returned.
As the the video playback is paused, the audio playback too is stopped.

PlayFfw Method

Purpose

Start playing back fast forward.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

```
long PlayFfw(long _window, long _speed)
```

Input parameters

Type	Name	Description
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long	_speed	Valid range [0 ... 512] power of 2. Any other value is rounded down to the nearest power of 2

Return

1 on success.

-1 if a generic error occurred.

Remarks

This method requires the window to be already in playback mode, so it only possible to call the PlayFfw() method only AFTER the Play() method has been called and no error was returned.

If _speed is 0 the playback goes one frame forward.

If _speed is 1 the playback is set at normal speed.

PlayRew Method

Purpose

Start playing backward.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

```
long PlayRew(long _window, long _speed)
```

Input parameters

Type	Name	Description
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long	_speed	Valid range [0 ... 512] power of 2. Any other value is rounded down to the nearest power of 2

Return

1 on success.

-1 if a generic error occurred.

Remarks

This method requires the window to be already in playback mode, so it only possible to call the PlayRew() method only AFTER the Play() method has been called and no error was returned.

If _speed is 0 the playback goes one frame backward.

If _speed is 1 the playback is set at normal speed.

PlayLocal Method

Purpose

Start to playback an extern file containing exported video footage in PSV format or a still image in SSF format.

Compatibility

Proxima: yes

Linearis: no

Spectiva: yes

Syntax

```
long PlayLocal(BSTR _nameString)
```

Input parameters

Type	Name	Description
BSTR	_nameString	Full path of image or video to playback.

Return

1 on success.

-1 if the file was not found.

-2 Error while reading header or header CRC error. Probably an unsupported file.

AddToMPList Method

Purpose

Add a window to the multiple playback windows list.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

```
long AddToMPList(long _window)
```

Input parameters

Type	Name	Description
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.

Return

1 on success.

-1 if the selected window is invalid.

Remarks

How does multiple playback work?

Multiple synchronized playback works using a list of windows to be visualized at the same time, keeping the playback time of each window congruent with the hour and date of a particular window, the MASTER WINDOW.

So, multiple playback needs 2 two steps to be completed before being started:

1. Multiple playback windows list have to be prepared before calling the Play() method. This list can be populated also with one bare element, but it can't be empty. Other windows can be attached to the list even after the playback has started. This list can be populated calling the AddToMPList() method. When creating this list for the first time, it's suggested to clear it calling the ClearMPList() method.
2. A master window has to be set before calling the Play() method. This can be done calling the SetMPMaster() method. The window set as master must be present in the multiple playback windows list, otherwise an error will be returned and multiple playback could not start.

As both single camera playback and multiple playback are started calling the Play() method, to switch from multiple playback back to single playback, the

window list has to be cleared, using ClearMPList().

The Multiple synchronized playback is a Cross-Connection function, so many different windows belonging to different connections (but always connections to Spectiva servers) can be added to the multiple playback window list.

IMPORTANT NOTE: each instance of the ActiveX control support one only multiple playback list!!!

The master window can be changed even when the playback has already started, simply calling the SetMPMaster() method. But if is set as master an invalid window (because not existing or not present in the multiple playback windows list) an OnWinStatusChange event will be triggered, and the parameter _reason will probably be MASTER_REMOVED. This very event will be triggered also if the master window is removed from the windows list.

See also

- Play() Method.
- RemoveFromMPList() Method.
- ClearMPList() Method.
- SetMPMaster() Method.
- GetMPMaster() Method.
- GetMPList() Method.
- OnWinStatusChange Event.

RemoveFromMPList Method

Purpose

Remove a window from the multiple playback windows list.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

```
long RemoveFromMPList(long _window)
```

Input parameters

Type	Name	Description
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.

Return

1 on success.

-1 if the selected window is invalid.

Remarks

If the removed window is the master window of the multiple playback, an OnWinStatusChange event will be triggered, and the parameter _reason will probably be MASTER_REMOVED.

See AddToMPList() method for VERY IMPORTANT remarks about multiple playback.

See also

Play() Method.

AddToMPList() Method.

ClearMPList() Method.

SetMPMaster() Method.

GetMPMaster() Method.

GetMPList() Method.

OnWinStatusChange Event.

ClearMPList Method

Purpose

Remove any window from they multiple play windows list.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

```
long ClearMPList()
```

Return

1 on success.

Remarks

This method has to be called to switch from multiple playback to single camera playback.

It's also suggested to clear the list before inserting the first camera in the list.

See AddToMPList() method for VERY IMPORTANT remarks about multiple playback.

See also

Play() Method.

AddToMPList() Method.

RemoveFromMPList() Method.

SetMPMaster() Method.

GetMPMaster() Method.

GetMPList() Method.

SetMPMaster Method

Purpose

Set a camera in the multiple playback windows list as Master camera.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

```
long SetMPMaster(long _window)
```

Input parameters

Type	Name	Description
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.

Return

1 on success.

-1 if the selected window is invalid.

Remarks

A master camera has to be set before starting multiple playback. If the multiple playback windows list is not empty, but no master has been set, the system won't be able to start playback, nor single, nor multiple.

See AddToMPList() method for VERY IMPORTANT remarks about multiple playback.

See also

Play() Method.

AddToMPList() Method.

RemoveFromMPList() Method.

ClearMPList() Method.

GetMPMaster() Method.

GetMPList() Method.

GetMPMaster Method

Purpose

Retrieve the master camera in the multiple playback windows list.

Compatibility

Proxima: no
Linearis: no
Spectiva: yes

Syntax

```
long GetMPMaster()
```

Return

The index of the master window, on success.
-1 if the master camera was not found.

Remarks

See AddToMPList() method for VERY IMPORTANT remarks about multiple playback.

See also

Play() Method.
AddToMPList() Method.
RemoveFromMPList() Method.
ClearMPList() Method.
GetMPMaster() Method.
GetMPList() Method.

GetMPList Method

Purpose

Fill a preallocated array of window indexes present in the multiple playback windows list.

Compatibility

Proxima: no
Linearis: no
Spectiva: yes

Syntax

```
long GetMPList (long FAR* _winHandles, long FAR* _count)
```

Input parameters

Type	Name	Description
long FAR*	_winHandles	Pointer to an empty array of (_count * sizeof(long)) byte.
long FAR*	_count	Number of DWORD

Return

1 on success.
-1 if the array was not correctly allocated.

Remarks

The array is NOT filled with real Windows handlers, but they're the internal window indexes.
See AddToMPList() method for VERY IMPORTANT remarks about multiple playback.

See also

Play() Method.
AddToMPList() Method.
RemoveFromMPList() Method.
ClearMPList() Method.
GetMPMaster() Method.
SetMPMaster() Method.

GetFindData Method

Purpose

Ask the DVR server for "Find" data to be sent.

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

```
long GetFindData(long connection, long cameras, long sector, long data);
```

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	cameras	A bit-field containing which cameras data are asked for. Use only the 16 less significant bits, where bit 1 stays for camera 1, bit 2 stays for camera 2, and so on.
long	sector	Number of the sector to be played back. See ChangeSectorWin for Constants values.
long	data	Pointer to a FindDataStruct structure (see Remarks for details)

Return

1 on success.

Remarks

To start a new find session, at first a GetFindData method has to be called, where a findDataStruc structure has to be filled AND THE FIELD sendData HAS TO BE 2.

```
struct FindDataStruct  
{  
    __int64 startTime;  
    __int64 endTime;  
    __int64 seconds; // time filter: Intervals far each other less than this are  
                    // considered as one bare interval.  
    unsigned long int sendData; // sendData = 2 for initial data request.  
};
```

When required data are ready to be sent to the client, an “OnFindData” event is triggered. Once a request has been sent, the first “OnFindData” event responses with a FindDataStruct containig precise data for each filed. This same structure can be used to forward a new GetFindData request, whose reply event will be another “onFindData” event, but this time it will contain a buffer with n couples of ___int64, where n is the number of recorded intervals.

See Also

onFindData event.



ConvertImageToBitmap Method

Purpose

Converts the last received Wavelet to bitmap and gives back informations about the image converted.

Compatibility

Proxima: yes

Linearis: no

Spectiva: no

Syntax

```
long ConvertImageToBitmap(long* _destBuffer, long* _date, long* _numSector,  
long* _numCamera, long* _local, long _connection);
```

Input parameters

Type	Name	Description
long*	_destBuffer	Return parameter. A buffer containing the last wavelet received converted to bitmap format.
long*	_date	Return parameter. A time_t representing the image date and hour
long*	_numSector	Return parameter. Number of the sector which the image belongs to.
long*	_numCamera	Return parameter. Camera number of the converted image.
long*	_local	Return parameter. 0 for remote connection, 1 for local playback.
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Return

1 on success.

ConvertImageToBitmapWinEx Method

Purpose

Converts the image shown in the selected window to bitmap and gives back informations about the image converted.

Compatibility

Proxima: no
Linearis: yes
Spectiva: yes

Syntax

```
long ConvertImageToBitmapWinEx(long* _destBuffer, long* _date, long*  
_numSector, long _window, long* _numCamera, long* _local, long _connection,  
long _width, long _height )
```

Input parameters

Type	Name	Description
long*	_destBuffer	Return parameter. A buffer containing the last image received converted to bitmap format.
long*	_date	Return parameter. A time_t representing the image date and hour
long*	_numSector	Return parameter. Number of the sector which the image belongs to.
long	_window	ID of the window showing the image we want to convert to bitmap.
long*	_numCamera	Return parameter. Camera number of the converted image.
long*	_local	Return parameter. 0 for remote connection, 1 for local playback.
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	_width	Desired width of the returned bitmap
long	_height	Desired height of the returned bitmap

Return

1 on success.



SaveBitmapWinToFile Method

Purpose

Convert the image in the given window to bitmap format and display the "Save File" dialog.

Compatibility

Proxima: no
Linearis: yes
Spectiva: yes

Syntax

long SaveBitmapWinToFile (long _window)

Input parameters

Type	Name	Description
long	_window	ID of the window showing the image we want to convert to bitmap.

Return

1 on success.
-1 on generic error.
-2 if a user right error occurred.

SaveBitmapWinToFileEx Method

Purpose

Convert the image in the given window to bitmap format and display the "Save File" dialog. Custom image size can be specified,

Compatibility

Proxima: no
Linearis: yes
Spectiva: yes

Syntax

long SaveBitmapWinToFileEx (long window, long width, long height, LPCTSTR defaultFileName)

Input parameters

Type	Name	Description
long	window	ID of the window showing the image we want to convert to bitmap.
long	width	Expected width of the output bitmap
long	height	Expected height of the output bitmap
LPCTSTR	defaultFileName	Default name of the output bitmap

Return

1 on success.
-1 on generic error.
-2 if a user right error occurred.

VideoQualityEnhanceWin Method

Purpose

Improves image quality by removing Mpeg4 and Enpacta compression artifacts.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

```
long VideoQualityEnhanceWin(long connection, long window,  
long status)
```

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long	status	Quality of the post processing

Return

1 on success.

Remarks

If "window" parameter is set to -1 post processing is applied to all the windows belonging to the selected connection.

If "window" parameter is different from -1, connection parameter is useless, so it's ignored.

If "connection" parameter and "window" parameter are set to -1, quality enhance is applied to all the windows.

Valid values for status parameter are:

noVideoQualityEnhanceStatus = 0

VideoQualityEnhanceStatusLow = 1

VideoQualityEnhanceStatusMedium = 2

VideoQualityEnhanceStatusHigh = 3

The higher the processing status parameter, the higher quality is achieved, the most CPU is used.

Default status is 0.

Using Wavelet or Enpacta as compression codec there are no differences between VideoQualityEnhanceStatusLow, VideoQualityEnhanceStatusMedium and VideoQualityEnhanceStatusHigh.



EnableMpegD1QualityEnhanceWin Method

Purpose

Improves image quality by removing unwanted motion artifacts .

Compatibility

Proxima: no
Linearis: yes
Spectiva: yes

Syntax

long EnableMpegD1QualityEnhanceWin(long connection, long window, long status)

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long	status	Status of the quality improvements. Set status to 0 to enable; Set status to 1 to disable.

Return

1 on success.

Remarks

If "window" parameter is set to -1 post processing is applied to all the windows belonging to the selected connection.
If "window" parameter is different from -1, connection parameter is useless, so it's ignored.
If "connection" parameter and "window" parameter are set to -1, quality enhance is applied to all the windows.

AUX AND ALARMS Methods

This group contains all the methods necessary to set and get informations and status of Auxes and Alarms.

- SendAlarm
- SendAux
- SendCameraAux
- ReceiveStatusAuxAlarm
- GetAlarmName

SendAlarm Method

Purpose

Notify to the DVR server to turn on or off the specified alarm.

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

```
long SendAlarm(long _connection, long _numAlarm, long _status)
```

Input parameters

Type	Name	Description
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	_numAlarm	Number of the alarm to turn on or off. Valid range[0 .. 15]
long	_status	Boolean set to 1 to turn on the alarm, 0 to turn it off.

Return

1 on success.
-1 if a generic error occurred.

SendAux Method

Purpose

Notify to the DVR server to turn on or off the specified AUX.

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

```
long SendAux(long _connection, long _numAux, long _status)
```

Input parameters

Type	Name	Description
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	_numAux	Number of the Aux to turn on or off. Valid range[0 .. 15]
long	_status	Boolean set to 1 to turn on the Aux, 0 to turn it off.

Return

1 on success.
-1 if a generic error occurred.

SendCameraAux Method

Purpose

Notify to the DVR server to turn on or off the specified AUX of the specified Camera.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: no

Syntax

long SendCameraAux(long connection, long, camera, long numAux, long status)

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	camera	Number of the camera whose AUX has to be set.
long	numAux	Number of the Aux to turn on or off. Valid range[0 .. 15]
long	status	Boolean set to 1 to turn on the Aux, 0 to turn it off.

Return

1 on success.

-1 if a generic error occurred.

ReceiveStatusAuxAlarm Method

Purpose

Ask the DVR server to send the activation status of AUX and alarms.

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

```
long ReceiveStatusAuxAlarm (long _connection);
```

Input parameters

Type	Name	Description
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Return

1 on success.
-1 if a generic error occurred.

Remarks

As the DVR server finish collecting the required informations, a OnAuxAlarmStatus event is triggered, and a 32 bits bit-field is returned, in which the 16 less significant bits contain the alarms status, while the most significant bits contain the aux status.

If the DVR is a Spectiva one, the ActiveX triggers the OnAlarmStatus and OnAuxStatus events and not the OnAuxAlarmStatus.

GetAlarmName Method

Purpose

Ask the DVR server the name of an alarm.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

```
BSTR GetAlarmName(long _connection, long _alarm);
```

Input parameters

Type	Name	Description
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	_alarm	The number of the alarm

Return

The name of the alarm on success.

A NULL BSTR if an error occurred.

AUDIO MANAGEMENT Methods

This group contains all the methods necessary to manage audio channels settings.

- SetAudioVolume
- SetBidirectionalAudioVolume

SetAudioVolume Method

Purpose

Set the volume of the desired audio channel in the specified connection.

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

```
long SetAudioVolume(long _connection, long _volume, long _channel)
```

Input parameters

Type	Name	Description
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	_volume	Volume value to set for the audio channel. Valid range [0 ... 100]; 0 means to turn off.
long	_channel	Number of the audio channel whose volume is to be set.

Return

1 on success.
-1 if a generic error occurred.

Remarks

When connecting to a DVR server whose audio channel support is limited to 1, the parameter _channel is always to be set to 0.

SetBidirectionalAudioVolume Method

Purpose

Set the volume of the TALK channel from client to server.

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

```
long SetBidirectionalAudioVolume(long _connection, long _volume, long  
_channel);
```

Input parameters

Type	Name	Description
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	_volume	Volume to be set for the TALK channel. Use 0 to turn it off. Valid Range [0 .. 100]
long	_channel	Number of the audio channel whose volume is to be set.

Return

0 on success.
-1 if a generic error occurred or the connection ID is invalid.
-2 Error creating audio recorder: hardware failure?

Remarks

Depending on the different audio board, it could sometimes happen that this method works as a binary switch to just turn on or off the TALK channel. In this case, the talk channel is turned off for volume value 0, while it could be active with the max volume for any other value.

DOME PILOTING Methods

This group contains all the methods necessary to pilot dome cameras .

- MoveDome
- ZoomDome
- IrisDome
- FocusDome
- SelectPresetDome
- SelectTourDome
- CaptureWindowsEvents
- EnableDomeControlWin

Move Method

Purpose

Move camera dome on X and/or Y axis.

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

```
long MoveDome(long _xSpeed, long _ySpeed, long _camera, long _connection);
```

Input parameters

Type	Name	Description
long	_xSpeed	Speed of dome movement on X (horizontal) axis. Valid range: [-100 .. +100]
long	_ySpeed	Speed of dome movement on Y (vertical) axis. Valid range: [-100 .. +100]
long	_camera	ID of camera Valid range: [0 .. 15]
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Return

1 on success.
-1 if a generic error occurred or the connection ID is invalid.
-2 if _xSpeed or _ySpeed parameter are out of range.
-3 if _camera is out of range.
-4 if the specified camera is not recognized as a dome.

Remarks

Set _xSpeed and _ySpeed to 0 to STOP the dome.

ZoomDome Method

Purpose

Zoom in/out the specified camera dome.

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

long ZoomDome(long _zoom, long _camera, long _connection)

Input parameters

Type	Name	Description
long	_zoom	Speed of zoom. Valid range [-100..100]; a positive value will zoom in a negative value will zoom out.
long	_camera	ID of camera Valid range: [0 .. 15]
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Return

1 on success.
-1 if a generic error occurred or the connection ID is invalid.
-2 if _zoom parameter is out of range.
-3 if _camera is out of range.
-4 if the specified camera is not recognized as a dome.

IrisDome Method

Purpose

Set iris value on camera dome.

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

```
long IrisDome(long _iris, long _camera, long _connection)
```

Input parameters

Type	Name	Description
long	_iris	Iris opening value. Valid range [-100..100]; a positive value will open the iris, a negative value will close it.
long	_camera	ID of camera Valid range: [0 .. 15]
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Return

1 on success.
-1 if a generic error occurred or the connection ID is invalid.
-2 if _iris parameter is out of range.
-3 if _camera is out of range.
-4 if the specified camera is not recognized as a dome.

FocusDome Method

Purpose

Set focus value on camera dome.

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

long FocusDome(long _focus, long _camera, long _connection)

Input parameters

Type	Name	Description
long	_focus	Focus value. Valid range [-100..100]; a negative value will focus far, while a positive value will focus near.
long	_camera	ID of camera Valid range: [0 .. 15]
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Return

1 on success.
-1 if a generic error occurred or the connection ID is invalid.
-2 if _focus parameter is out of range.
-3 if _camera is out of range.
-4 if the specified camera is not recognized as a dome.

Remarks

Set _focus to 0 to STOP the dome focus moving.

SelectPresetDome Method

Purpose

Order the dome to move to specified preset.

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

```
long SelectPresetDome(long _preset, long _camera, long _connection)
```

Input parameters

Type	Name	Description
long	_preset	Number of the preset to move to.
long	_camera	ID of camera Valid range: [0 .. 15]
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Return

1 on success.
-1 if a generic error occurred or the connection ID is invalid.
-2 if _preset parameter is out of range.
-3 if _camera is out of range.
-4 if the specified camera is not recognized as a dome.

SelectTourDome Method

Purpose

Order the dome to start or stop the specified tour.

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

```
long SelectTourDome(long _tour, long _camera, long _connection, long _on);
```

Input parameters

Type	Name	Description
long	_tour	Number of the preset to move to.
long	_camera	ID of camera Valid range: [0 .. 15]
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	_on	Boolean. Set to 1 to start the tour, set to 0 to stop it.

Return

1 on success.
-1 if a generic error occurred or the connection ID is invalid.
-2 if _tour parameter is out of range.
-3 if _camera is out of range.
-4 if the specified camera is not recognized as a dome.

CaptureWindowsEvents Method

Purpose

Use this method to switch between manual and automatic dome piloting. If automatic is selected, all the mouse related events will be trapped by the ActiveX control and NOT passed to the host application.

Compatibility

Proxima: no
Linearis: yes
Spectiva: no

Syntax

```
long CaptureWindowsEvents(long _status);
```

Input parameters

Type	Name	Description
long	_status	BOOLEAN: 1: enable automatic dome piloting 2: disable automatic dome piloting

Return

1 on success.

EnableDomeControlWin Method

Purpose

Call this method to enable or disable the dome piloting on the selected window

Compatibility

Proxima: no

Linearis: yes

Spectiva: yes

Syntax

```
long EnableDomeControlWin(long _window, long _status);
```

Input parameters

Type	Name	Description
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long	_status	Boolean: 1 to enable, 0 not to disable dome piloting on the selected window,

Return

1 on success.

-1 if a generic error occurred or the window ID is invalid.

SPOT MONITOR Methods

This group contains all the methods necessary to manage Spot Monitors, in normal and in custom mode, where supported.

- StartCustomSP
- StopCustomSP
- SetCustomCameraSP
- SetCustomDataSP
- ResetCustomDataSP

StartCustomSP Method

Purpose

Set the specified monitor to work in custom mode. Nothing will be shown until a SetCustomCameraSP or SetCustomDataSP method will be called.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: no

Syntax

```
long StartCustomSP(long connection, long spot_monitor)
```

Input parameters

Type	Name	Description
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	spot_monitor	Number of spot monitor selected.

Return

1 on success.

Remarks

While in custom mode it is possible to view a single camera using SetCustomCameraSP, or it is possible to show a cycle using SetCustomDataSP.

StopCustomSP Method

Purpose

Exit the specified monitor from the custom mode.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: no

Syntax

long StartCustomSP(long connection, long spot_monitor)

Input parameters

Type	Name	Description
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	spot_monitor	Number of spot monitor selected.

Return

1 on success.

Remarks

Once out of the custom mode it will be not possible to use SetCustomCameraSP, SetCustomDataSP and ResetCustomDataSP, but it will be possible to call SetSpotMonitor and ResetSpotMonitor.

SetCustomCameraSP Method

Purpose

Set a camera to be displayed on the specified spot monitor while in custom mode.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: no

Syntax

```
long SetCustomCameraSP(long connection, long spot_monitor, long camera, long show_text);
```

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	spot_monitor	Number of spot monitor selected.
long	camera	Number of the camera to be shown. Valid range [0..15]
long	show_text	Boolean: 1 to show overlay text on the spot monitor, 0 not to show any text

Return

1 on success.

Remarks

Before using SetCustomCameraSP it is necessary to enter in custom mode by calling StartCustomSP method.

SetCustomDataSP Method

Purpose

Set a camera cycle to be displayed on the specified spot monitor while in custom mode.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: no

Syntax

```
long SetCustomDataSP(long connection, long spot_monitor, long cameras, long show_text, long seq_time);
```

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	spot_monitor	Number of spot monitor selected.
long	cameras	A bit-field containing in the less significant bits which cameras have to be displayed
long	show_text	Boolean: 1 to show overlay text on the spot monitor, 0 not to show any text
long	seq_time	Seconds of visualization of each camera in the cycle.

Return

1 on success.

Remarks

Before using SetCustomDataSP it is necessary to enter in custom mode by calling StartCustomSP method.

ResetCustomDataSP Method

Purpose

Reset the custom cycle. Nothing will be shown in the Spot monitor.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: no

Syntax

long ResetCustomDataSP(long connection, long spot_monitor)

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	spot_monitor	Number of spot monitor to be reset.

Return

1 on success.

Remarks

Before using ResetCustomDataSP it is necessary to enter in custom mode by calling StartCustomSP method.

EXPORT Methods

This group contains all the methods necessary to export A/V streams to AVI or PSV files.

This functions usually manage the ActiveX as a provider of audio, video and informations streams to be exported to A/V files using Insignis ArchiveExport ActiveX control.

- ExportStart
- GetExportInfo
- ExportStop
- ExportPause
- ExportResume

ExportStart Method

Purpose

Ask the server to start sending packets of images for export purpose from and to the specified dates.

Compatibility

Proxima: no
Linearis: yes
Spectiva: yes

Syntax

```
long ExportStart(long connection, long * startDate, long * endDate, long camera, long sector, long maxBufSize, long maxImages);
```

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long*	startDate	Pointer to an _int64 containing the starting hour and date of the fragment to export.
long*	endDate	Pointer to an _int64 containing the ending hour and date of the fragment to export.
long	camera	Number of the camera to export.
long	sector	Number of the sector to export. 0 = Prime; 1 = Time Lapse; 2 = Alarms.
long	maxBufSize	Expressed in bytes, is the max size of the image buffer returned. 0 = automatic.
long	maxImages	Maximum number of frames to export. Valid range: [1 .. 10] Maximal performed can be achieved setting this value between 6 and 10.

Return

0 on success.
-1 if the connection ID is invalid.

Remarks



As this method is called, data will be returned via the OnExportData event. When data are over, or if an error occurs, an OnExportStatus event will be triggered. No more than one export stream can be started for each connection.

See Also

GetExportInfo method.
ExportStop methods.
OnExportData event.
OnExportStatus event.



GetExportInfo Method

Purpose

Ask the server for informations about the export process.

Compatibility

Proxima: no
Linearis: yes
Spectiva: yes

Syntax

```
long GetExportInfo(long connection, long * startDate, long * endDate, long camera, long sector, long infoType);
```

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long*	startDate	Pointer to an _int64 containing the starting hour and date of the fragment to export.
long*	endDate	Pointer to an _int64 containing the ending hour and date of the fragment to export.
long	camera	Number of the camera to export.
long	sector	Number of the sector to export. 0 = Prime; 1 = Time Lapse; 2 = Alarms.
long	infoType	

Return

a non-negative Stream ID on success.
-1 if the connection ID , the camera or the sector is invalid.

ExportStop Method

Purpose

Abort the export process.

Compatibility

Proxima: no

Linearis: yes

Spectiva: yes

Syntax

```
long ExportStop(long connection, long streamId);
```

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	streamId	The Stream ID returned by ExportStart() method.

Return

0 on success.

-1 if the connection ID , the camera or the sector is invalid.

ExportPause Method

Purpose

Call this method to pause a running export process.

Compatibility

Proxima: no

Linearis: yes

Spectiva: yes

Syntax

```
long ExportPause(long connection, long streamId);
```

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	streamId	The Stream ID returned by ExportStart() method.

Return

0 on success.

-1 if the connection ID , the camera or the sector is invalid.

Remarks

Call the ResumeExport() method to resume a pause export process

ExportResume Method

Purpose

Resume a paused export process.

Compatibility

Proxima: no

Linearis: yes

Spectiva: yes

Syntax

```
long ExportResume(long connection, long streamId);
```

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	streamId	The Stream ID returned by ExportStart() method.

Return

0 on success.

-1 if the connection ID , the camera or the sector is invalid.

TEXT INSERTION Methods

This group contains all the methods to setup the display of text insertion data and creation of text insertion data to be inserted in the recordings archive.

- SetTextInsertionOptions
- SetTextInsertionOptionsEx
- GetTextInsertionOptions
- GetTextInsertionOptionsEx
- SendDeviceTextInsertionData

SetTextInsertionOptions Method

Purpose

This method has to be used to setup the display options for the text insertion feature. This options apply to all the windows belonging to the selected connection.

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

```
long SetTextInsertionOptions(long connection, long view_live, long view_play,  
long top, long left, long bottom, long right)
```

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	view_live	Boolean: 1 to show text acquired from any text insertion device in live window of the given connection, 0 not to show any text in live.
long	view_play	Boolean: 1 to show text acquired from any text insertion device in playback windows of the given connection, 0 not to show any text in play.
long	top	Upper ordinates of the text display box. Valid Range: [0 .. bottom)
long	left	Left abscissas of the text display box. Valid Range: [0 .. right)
long	bottom	Lower ordinates of the text display box. Valid Range: (top .. 70]
long	right	Right abscissas of the text display box. Valid Range: (left .. 84]

Return

1 on success.
0 if any of the coordinates is invalid.
-1 if the connection ID is invalid.

Remarks



All coordinates about the text visualization box should be referred to a grid of 84x70 cells, not regarding the window resolution. This setup data are common for every window (both live and play) of the specified connection.

See Also

GetTextInsertionOptions method.



SetTextInsertionOptionsEx Method

Purpose

This method has to be used to setup the display options for the text insertion feature. This options only apply to the selected camera,

Compatibility

Proxima: no
Linearis: yes
Spectiva: yes

Syntax

long SetTextInsertionOptions(long connection, long camera, long id, long view_live, long view_play, long top, long left, long bottom, long right, long duration)

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	camera	Number of the camera
long	id	A valid text insertion device ID
long	view_live	Boolean: 1 to show text acquired from any text insertion device in live window of the given connection, 0 not to show any text in live.
long	view_play	Boolean: 1 to show text acquired from any text insertion device in playback windows of the given connection, 0 not to show any text in play.
long	top	Upper ordinates of the text display box. Valid Range: [0 .. bottom)
long	left	Left abscissas of the text display box. Valid Range: [0 .. right)
long	bottom	Lower ordinates of the text display box. Valid Range: (top .. 70]
long	right	Right abscissas of the text display box. Valid Range: (left .. 84]
long	duration	Duration in seconds of the overlay text.

Return

1 on success.



0 if any of the coordinates is invalid.
-1 if the connection ID is invalid.

Remarks

All coordinates about the text visualization box should be referred to a grid of 84x70 cells, not regarding the window resolution.

See Also

GetTextInsertionOptionsEx method.



GetTextInsertionOptions Method

Purpose

Call this method to get actual settings for text insertion for the given connection.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: no

Syntax

```
long GetTextInsertionOptions(long connection, long* view_live, long* view_play,  
long* top, long* left, long* bottom, long* right)
```

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long*	view_live	Boolean: 1 means that text acquired from any text insertion device is shown in live windows of the given connection, 0 means that no text is shown in live.
long*	view_play	Boolean: 1 means that text acquired from any text insertion device is shown in playback windows of the given connection, 0 means that no text is shown in playback.
long*	top	Upper ordinates of the text display box.
long*	left	Left abscissas of the text display box.
long*	bottom	Lower ordinates of the text display box.
long*	right	Right abscissas of the text display box.

Return

1 on success.

Remarks

All coordinates about the text visualization box are referred to a grid of 84x70 cells, not regarding the window resolution.
This setup data are common for every window (both live and play) of the specified connection.

See Also

SetTextInsertionOptions method.



GetTextInsertionOptionsEx Method

Purpose

Call this method to get actual settings for text insertion for the given connection.

Compatibility

Proxima: no
Linearis: yes
Spectiva: yes

Syntax

```
long GetTextInsertionOptions(long connection, long* view_live, long* view_play,  
long* top, long* left, long* bottom, long* right)
```

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	camera	Number of the camera
long	id	A valid text insertion device ID
long*	view_live	Boolean: 1 means that text acquired from any text insertion device is shown in live windows of the given connection, 0 means that no text is shown in live.
long*	view_play	Boolean: 1 means that text acquired from any text insertion device is shown in playback windows of the given connection, 0 means that no text is shown in playback.
long*	top	Upper ordinates of the text display box.
long*	left	Left abscissas of the text display box.
long*	bottom	Lower ordinates of the text display box.
long*	right	Right abscissas of the text display box.
long*	duration	Duration in seconds of the overlay text.

Return

1 on success.

Remarks

All coordinates about the text visualization box are referred to a grid of 84x70 cells, not regarding the window resolution.

SendDeviceTextInsertionData Method

Purpose

This method allows to insert text insertion data in the storage.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: no

Syntax

SendDeviceTextInsertionData(long connection, long id, long cameras, BSTR text)

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	id	ID of the device which is used to create the text. This ID must be a valid text insertion device id created in the server setup and marked as "Ethernet device"
long	cameras	A bit-field containing which camera data are to be stored for, where bit 1 stays for camera 1, bit 2 stays for camera 2, and so on.
BSTR	text	Text to be inserted in the archive (and/or shown in the text box)

Return

1 on success.

Remarks

Max text size is 200 characters

MANUAL RECORDING Methods

This group contains all the methods to handle the manual recording.

- StartManualRec
- StopManualRec
- ResetManualRec
- GetRecordingMode

StartManualRec Method

Purpose

This method starts the manual recording of the specified camera on the specified sector.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

long StartManualRec(long connection, long camera, long sector)

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	camera	Number of the camera to force recording.
long	sector	Number of the sector on which record video data.

Return

1 on success.

Remarks

Use following constants value for sector parameter:

Sector	Value
PRIME	0
TIME LAPS	1
ALARMS	2

StopManualRec Method

Purpose

This method stops the manual recording of the specified camera on the specified sector; this means that the recording is stopped in general, regardless to its standard scheduler.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

long StopManualRec(long connection, long camera, long sector)

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	camera	Number of the camera to stop recording on the specified sector.
long	sector	Number of the sector.

Return

1 on success.

Remarks

Use following constants value for sector parameter:

Sector	Value
PRIME	0
TIME LAPS	1
ALARMS	2

ResetManualRec Method

Purpose

This method stops the manual recording of the specified camera on the specified sector; this doesn't mean that the recording is stopped in general, but the system is switched back to its standard scheduler.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

long StopManualRec(long connection, long camera, long sector)

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	camera	Number of the camera to remove from the forced recording on the specified sector.
long	sector	Number of the sector to remove from the forced recording.

Return

1 on success.

Remarks

Use following constants value for sector parameter:

Sector	Value
PRIME	0
TIME LAPS	1
ALARMS	2

GetRecordingMode Method

Purpose

Asks the server the status of the recording on the specified camera and sector. An OnRecordingMode event is triggered as the answer is received.

Compatibility

Proxima: yes
Linearis: yes
Spectiva: no

Syntax

long GetRecordingMode(long connection, long camera, long sector)

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	camera	Number of the camera to remove from the forced recording on the specified sector.
long	sector	Number of the sector to remove from the forced recording.

Return

1 on success.

Remarks

The return value is NOT the status of the recording. This is given by an "OnRecordingMode" event.

Use following constants value for sector parameter:

Sector	Value
PRIME	0
TIME LAPS	1
ALARMS	2

See Also

OnRecordingMode event.



PANORAMA VIEWS Methods

This group contains all the methods to handle the Panorama Views.

- SetPanoramaWin
- DestroyPanoramaWin
- AddWinToPanorama
- RemoveWinFromPanorama
- SetWinPanoramaOptions
- GetWinFromPoint

SetPanoramaWin Method

Purpose

Use this method to transform any existing GDI object derived from TWindows into a panorama-view drawing area.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

```
long SetPanoramaWin(long handle)
```

Input parameters

Type	Name	Description
long	handle	Handle of an existing GDI object to be transformed into a panorama window

Return

The ID of the panorama-view created.

DestroyPanoramaWin Method

Purpose

Call this method to free the GDI object whose ID is specified from the assignment to panorama-view.

Compatibility

Proxima: no
Linearis: no
Spectiva: yes

Syntax

```
long DestroyPanoramaWin(long handle)
```

Input parameters

Type	Name	Description
long	handle	ID of a panorama-view returned by a previous call of the SetPanoramaWin() method.

Return

1 on success.

Remarks

This method does NOT destroy the GDI object itself, only the association with a panorama-view.

AddWinToPanorama Method

Purpose

Call this method to make a window member of a panorama-view.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

```
long AddWinToPanorama(long window, long panorama_window)
```

Input parameters

Type	Name	Description
long	window	ID of a visualization window to insert in the panorama-view. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long	panorama_window	ID of a panorama-view returned by a previous call of the SetPanoramaWin() method.

Return

1 on success.

RemoveWinFromPanorama Method

Purpose

Use this method to remove the specified window from a panorama-view.

Compatibility

Proxima: yes

Linearis: no

Spectiva: no

Syntax

```
long RemoveWinFromPanorama(long window, long panorama_window)
```

Input parameters

Type	Name	Description
long	window	ID of the visualization window to be removed from the panorama-view. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long	panorama_window	ID of a panorama-view returned by a previous call of the SetPanoramaWin() method.

Return

1 on success.

SetWinPanoramaOptions Method

Purpose

Call this method to set the appropriate topographic settings of a camera belonging to a panorama-view.

Compatibility

Proxima: no
Linearis: no
Spectiva: yes

Syntax

```
long SetWinPanoramaOptions(long window, long panorama_window, long x1, long y1, long x2, long y2, long x3, long y3, long x4, long y4, long zorder)
```

Input parameters

Type	Name	Description
long	window	ID of the visualization window to be removed from the panorama-view. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long	panorama_window	ID of a panorama-view returned by a previous call of the SetPanoramaWin() method.
long	x1	Coordinates of the upper-left corner
long	y1	Coordinates of the upper-left corner
long	x2	Coordinates of the upper-right corner
long	y2	Coordinates of the upper-right corner
long	x3	Coordinates of the lower-right corner
long	y3	Coordinates of the lower-right corner
long	x4	Coordinates of the lower-left corner
long	y4	Coordinates of the lower-left corner
long	zorder	Valid range: [-256...256] Use negative values to send back a window 1 or 2 or 3 (...) levels. Use positive values to bring the window forward.

Return

1 on success.

GetWinFromPoint Method

Purpose

This method returns the ID of the topmost window present at the specified coordinates.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

```
long GetWinFromPoint (long panorama_window, long x, long y)
```

Input parameters

Type	Name	Description
long	panorama_window	ID of a panorama-view returned by a previous call of the SetPanoramaWin() method.
long	x	Abscissas of the point
long	y	Ordinates of the point

Return

ID of the topmost windows present at the coordinates.

EVENTS

The following events are provided by the event dispatch interface of the ActiveX control:

Event	Description
WatchDog	Triggered when the watchdog message is received from the DVR server.
SetupError	The setup received was corrupted or not compatible with the client.
OnVideoError	An error has occurred trying to show video.
BlockingEvent	Entering or exiting in a blocking operation.
PauseEvent	DEPRECATED.
CamerasReceived	After that the ActiveX received the setup structure this event is triggered to pass general infos about all connected cameras.
OnCreate	The first event triggered as the ActiveX object is created.
OnDoubleClick	The user has double-clicked on a camera window.
OnConnection	Replay for a connection request.
OnAlarm	The server notify that an alarm has changed.
OnAux	The server notify that an AUX has changed.
OnAuxAlarmStatus	The status of all alarm and AUX is sent to client.
OnCameraChanged	In mono mode, will be displayed an image of a camera different from the actual one.
OnBuffering	The first fragment of the first image of a stream is arrived, but we're still waiting for the other chunks.
OnDomesStatus	Notify which camera is a camera and which is a dome.
OnPlayStatus	After setup is received, notify if the playback is possible or not.
OnRecording	Recording of the specified camera has started or stopped.
OnDarkening	Darkening detected on the specified camera.
OnDateSync	Notify the exact hour and date of the server.
OnLoadingLocalPlay	DEPRECATED.
OnAuxAlarmStatusEx	DEPRECATED
OnAlarmsAvailable	Return the availability of alarms connector, both physical or logical.
OnAuxAvailable	DEPRECATED
OnCamerasDisconnected	Notify which camera is active but disconnected.
OnCamerasTooLarge	Notify which camera is active but the image is too large to be recorded or visualized.
OnFindData	Return a buffer of A/V streams fragments available.
OnStreamInfo	Periodically sends date and hour of the last image received.
OnAuxStatus	Return the status of up to 128 auxes. USB extension board IS supported.
OnAlarmStatus	Return the status of up to 128 alarms. USB extension board IS supported.
OnAuxAvailableEx	Return the availability of up to 128 AUX connector. USB extension board IS supported.

Event	Description
OnStreamImage	INTERNAL USE.
OnSectorsAvailable (proxima)	After that the ActiveX received the setup structure this event is triggered to pass general infos about all connected cameras.
OnSectorsAvailable (spectiva)	Return which camera is enabled to record on which sector.
OnWinStatusChange	Notify why a window has passed from play to live (and viceversa).
OnExportInfo	INTERNAL USE
OnExportData	Return data required to export.
OnExportStatus	Return the status of the export process.
OnCameraAuxAvailable	Return a bitfield of available AUXes for each camera.
OnCameraAuxStatus	Return a bitfield of the status of the AUXes of each camera.
OnCameraAux	Return the status of a changed aux of a camera.
OnRecordingMode	Returns the recording status of the specified camera and sector.
OnAlarmEx	Returns extended informations about an alarm.
OnNeedCertificate	When certification is needed, data verification is requested.
OnNeedCertificateBuffer	When certification is needed, data verification is requested, specifying the serial of the certificate to use.
OnNeedCertificatePlay	When certification is needed to display encrypted video footage, data verification is requested, specifying the serial of the certificate to use.
OnSetPriorityCamera	Notify when a priority camera is set.
OnAudioChannelsOnEx	Notify which audio channel(s) is associated to a camera.
OnChannelsAudioAvailableEx	Notify which audio channels are available on the server.
OnTalkChannelAvailable	INTERNAL USE
OnExportTIData	Return text insertion data required to export.
OnClick	Notify that the mouse left button has been clicked.
OnAudioChannelsOn	Notify which audio channel(s) is associated to a camera.
AudioChannelsOn	Notify which audio channel(s) is associated to a camera.
OnAudioChannelsAvailable	Notify which audio channels are available on the server.
OnAlarmsAvailable2	Return the availability of alarms connector, both physical or logical.
OnAlarmStatus2	Return the status of up to 128 alarms. USB extension board IS supported.
OnAlarmEx2	Returns extended informations about an alarm.
OnAuxStatus2	Return the status of up to 128 auxes. USB extension board IS supported.
OnAuxAvailableEx2	Return the availability of up to 128 AUX connector. USB extension board IS supported.
OnCameraAuxAvailable2	Return available AUXes for each camera.
OnCameraAuxStatus2	Return the status of the AUXes of each camera.
OnStreamInfo2	Periodically sends date and hour of the last image received.

WatchDog Event

Purpose

Notify that the watchdog message has been received from the server. This means that the connection is still alive.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

WatchDog(long _connection)

Input parameters

Type	Name	Description
long	_connection	ID of server connection for which the watchdog message is sent. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

SetupError Event

Purpose

The setup received was corrupted or (more probably) the setup version is not compatible with the client.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

SetupError (long _connection)

Input parameters

Type	Name	Description
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

OnVideoError Event

Purpose

Notify that the watchdog message has been received from the server. This means that the connection is still alive.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

OnVideoError (long _error)

Input parameters

Type	Name	Description
long	_error	Error code: 0 = Error initializing graphic engine. Common when using ProximaWeb ActiveX release 3.0.0.26 with a 16 bit color depth display. 1 = Other error.

BlockingEvent Event

Purpose

Notify that the ActiveX control is entering or exiting in a blocking operation. While in a blocking operation no other event can be triggered, and no method can be performed.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: no

Syntax

BlockingEvent(long _blocking)

Input parameters

Type	Name	Description
long	_blocking	Block Status: 1 = Actually entering into the blocking code fragment. 0 = The blocking operation has just terminated.

Remarks

While in a blocking operation, no other event will be triggered before another BlockingEvent(0) is raised to advise that the blocking operation has finished. If a method has been called in the meanwhile it is not guaranteed to be correctly executed.

CamerasReceived Event

Purpose

Notify that the ActiveX has received a new setup file from the server, then a string is passed containing basic data about server name and cameras name.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

CamerasReceived(BSTR _names, long _position)

Input parameters

Type	Name	Description
BSTR	_names	A formatted string containing server and camera names . See Remarks for details.
long	_position	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Remarks

_names is a string formatted in this way:

Server name

Marker (*)

Camera 1 name.

Marker (*)

Camera 2 name.

Marker (*)

Camera 3 name.

Marker (*)

Camera 4 name.

Marker (*)

...

Camera 16 name.

Marker (*)

If the first character of a camera name is an exclamation mark (!) it means that

the camera is disconnected



OnCreate Event

Purpose

This event is triggered as soon as the ActiveX control is created. So this is always the first event triggered.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

OnCreate()

Remarks

When this event is received means that methods can be called, otherwise they would be refused.

OnDoubleClick Event

Purpose

The OnDoubleClick event indicates that the local user has double-clicked a window in which the ActiveX was drawing an A/V stream.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

OnDoubleClick(long _window)

Input parameters

Type	Name	Description
long	_window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.

OnConnection Event

Purpose

Notify a connection request to the server, replaying if the request has been accepted or refused, and the error eventually raised.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

OnConnection(long _status, long _connection)

Input parameters

Type	Name	Description
long	_status	Connection error code. See remarks for code details.
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Remarks

OnConnection(long _status, long _connection)

_status value	description
0	Disconnected by server.
1	Valid connection.
2	Unable to find server.
3	Wrong user name or password.
4	Wrong or missing license.
5	Black-listed client.
6	Wrong setup version.
7	Too many users connected.

OnAlarm Event

Purpose

Notify the status of an alarm changed on the server.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

OnAlarm(long _numAlarm, long _status, long _connection)

Input parameters

Type	Name	Description
long	_numAlarm	Number of the changed alarm.
long	_status	Boolean: 1 for alarm active, 0 for alarm not active.
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

OnAlarmAvailableEx Event

Purpose

Returns the availability of Alarms in the given server. This event is triggered as the connection is established, and every time the setup is modified. USB Extension board is supported.

Compatibility

Proxima: yes, since release 3.05.00

Linearis: yes

Spectiva: no

Syntax

OnAlarmAvailableEx(long* _status, long _size, long _connection)

Input parameters

Type	Name	Description
long*	_status	Pointer to bitfield of _size bytes, organized into DWORD (32 bits). So _status[0] contains availability of alarms from 0 to 31, _status[1] contains availability of alarms from 32 to 63, and so on. 1 stands for "available", 0 for "not available".
long	_size	Size in bytes of the data structure pointed at _status.
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Syntax

In Proxima release 3.05.00 one only DWORD is returned, as 32 alarms are supported, so _size contains 4.

OnAux Event

Purpose

Notify the status of an AUX changed on the server.

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

OnAux(long _numAlarm, long _status, long _connection)

Input parameters

Type	Name	Description
long	_numAux	Number of the changed AUX.
long	_status	Boolean: 1 for AUX active, 0 for AUX not active.
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

OnAuxAlarmStatus Event

Purpose

The server notify to the client the status of ALL Proxima alarms and AUXes. In case this event is triggered in Spectiva ActiveX control, only data relative to the first 16 AUXes and Alarms are given.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: DEPRECATED

Syntax

OnAuxAlarmStatus(long _status, long _connection)

Input parameters

Type	Name	Description
long	_status	A bit-field containing the status of all alarms and AUXes. See remarks for details
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Remarks

_status parameter structure: the 32 bit parameter is a bit-field in which the lower 16 bits represent the activation status of alarms (1 for active, 0 inactive), while the higher 16 bits represent the activation status of AUXes (1 for active, 0 inactive)

See also

OnAuxStatus event.

OnAlarmStatus event.

OnCameraChanged Event

Purpose

Visualizing streams in mono mode, has arrived an image belonging to a camera different from the actual one.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

OnCameraChanged(long _camera, long _connection)

Input parameters

Type	Name	Description
long	_camera	Number of the camera that will be displayed from now on.
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Remarks

This event is triggered only when using one bare window for displaying A/V streams and the image arrived belongs to a camera different from the one actually displayed. This means that, as a consequence of SetCamera() method call, the first image of the new camera has arrived.

OnBuffering Event

Purpose

The first fragment of the first image of a stream is arrived, but we're still waiting for the other chunks.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

OnBuffering(long _value, long _connection)

Input parameters

Type	Name	Description
long	_value	Boolean: 1 for buffering beginning, 0 for buffering terminated.
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Remarks

This event is useful in case of very slow connections (56Kb or slower) where the waiting time before receiving a complete first frame could last some seconds. Suppose the connection speed is single line ISDN (64Kbps) and image quality is high, and no differential transmission is active, a single full frame could take more than 10 seconds. The use of this event could be useful to advert the user that the connection is working fine, and the image is arriving, but it's still necessary to wait a moment.

OnDomesStatus Event

Purpose

Notify to the client which camera is configured as a dome, and which camera is a normal camera.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

OnDomesStatus(long _cameras, long _connection)

Input parameters

Type	Name	Description
long	_cameras	A bit-field containing in the lower 16 bits a flag indicating the dome presence for each camera. 1 stands for "camera is a dome", 0 for "camera is not a dome"
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

OnPlayStatus Event

Purpose

After setup is received this event is triggered to notify to the client whether the playback is possible or not.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

OnPlayStatus(long _status, long _connection)

Remarks

The playback may not be available if no recorded material is present, or if the connected user is not allowed to view it.

Input parameters

Type	Name	Description
long	_status	Boolean: 1 for playback enabled, 0 for playback disabled.
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

OnRecording Event

Purpose

Notify that the recording of the specified camera has started or stopped.

Compatibility

Proxima: yes

Linearis: no, view OnRecordingMode

Spectiva: yes

Syntax

OnRecording(long _camera, long _status, long _connection)

Input parameters

Type	Name	Description
long	_camera	Number of the camera in which has changed the recording status
long	_status	Boolean: 1 if the recording has started, 0 if it has stopped.
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Remarks

This event is triggered even if the specified camera is not visualized. After this event, an OnDarkening event is always triggered.

OnDarkening Event

Purpose

Notify that darkening on the specified camera has been detected, or it is no more detected.

Compatibility

Proxima: yes

Linearis: no

Spectiva: yes

Syntax

OnDarkening(long _camera, long _status, long _connection)

Input parameters

Type	Name	Description
long	_camera	Number of the camera in which has changed the darkening status
long	_status	Boolean: 1 if the darkening has been detected, 0 if it is no more detected.
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Remarks

This event is triggered even if the specified camera is not visualized. Before this event, an OnRecording event is always triggered.

OnDateSync Event

Purpose

Notify to the client the exact hour and date of the server.

Compatibility

Proxima: yes

Linearis: yes ,since release 1.06.00

Spectiva: yes

Syntax

OnDateSync(long _hour, long _min, long _sec, long _day, long _month, long _year, long _connection)

Input parameters

Type	Name	Description
long	_hour	Hour of the server
long	_min	Minutes of the server
long	_sec	Seconds of the server
long	_day	Day of the server
long	_month	Month of the server
long	_year	Year of the server
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Remarks

No information is given about timezone or daylight. Hour and date are given as shown on the server

OnAudioChannelsOn Event

Purpose

Notify which audio channels are associated to the given camera.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: no

Syntax

OnAudioChannelsOn(long _connection, long _channels, long _camera)

Input parameters

Type	Name	Description
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	_channels	A 32 bit bit-field. Less significant bit is channel 1, the most significant is channel 32. 1 stands for "this channel IS associated to the camera". 0 stands for "this channel IS NOT associated to the camera". Spectiva supports up to 4 channels associated to a camera, while Proxima supports one only channel.
long	_camera	Number of the camera.

OnAlarmsAvailable Event

Purpose

Return the availability of the 128 alarms.

Compatibility

Proxima: no
Linearis: yes
Spectiva: yes

Syntax

OnAlarmsAvailable(long * _status, long _connection)

Input parameters

Type	Name	Description
long*	_status	Pointer to a 4 DWORD array containing a bit-field of infos about the alarms availability. See remarks for details.
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Remarks

The 4 DWORD array is build up this way:
DWORD 0: 0..31 = alarms 0..31
DWORD 1: 0..31 = alarms 32..63
DWORD 2: 0..31 = alarms 64..95
DWORD 3: 0..31 = alarms 96..127
Bit set (1) means state is ON.
Bit clear (0) means state is OFF.

OnCamerasDisconnected Event

Purpose

Notify which cameras are set as active but are actually disconnected.

Compatibility

Proxima: yes
Linearis: yes
Spectiva: yes

Syntax

OnCamerasDisconnected(long _status, long _connection)

Input parameters

Type	Name	Description
long	_status	A bit-field containing int the 16 less significant bits which camera is active but disconnected (where bit 0 is camera 1... bit 15 is camera 16)
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

OnCamerasTooLarge Event

Purpose

Notify which camera is not being visualized (nor recorded) because the image is too large (exceeding 100 KB / frame).

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

OnCamerasTooLarge(long _status, long _connection)

Input parameters

Type	Name	Description
long	_status	A bit-field containing int the 16 less significant bits which camera is in "Too Large" status (where bit 0 is camera 1... bit 15 is camera 16)
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

OnFindData Event

Purpose

As a request of a GetFindData() an OnFindData event is triggered containing infos about the recorded material. See remarks for further details.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

OnFindData(long connection, long camera, long* info, long countIntervals, long* data, long finished)

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	camera	Camera for which the find data are given.
long*	info	Pointer to a findDataStruc record containing
long	countIntervals	Number of intervals returned
long*	data	Pointer to a [countIntervals] couples of _int64 representing each interval beginning and finish time.
long	finished	Unused.

Remarks

The FIND operation begins with a GetFindData method where the last parameter is a pointer to a FindDataStruc record (see GetFindData method for details) , where is asked a generic time interval and the sendData field is set to 2, constant value for data request.

The answer to this request is a OnFindData event where the field info contains a pointer to another FindDataStruc record filled with precise beginning and ending hour, the applied filter and the sendData filed set yet to the correct value for extensive data request.

This very returned record is to be used to perform a second GetFindData request. At this point another OnFindData event will be triggered, where the parameter "countIntervals" will contains the real number of intervals returned. The parameter "data" contains a pointer to an array of [countIntervals] records composed by 2 _int64, where the first one is the beginning hour of an interval, the second one is the ending time of that same interval. The "info" field contains a pointer to a FindDataStruc record which *SHOULD* be equal to the other previously returned. In case of difference, the fist received is the valid one.

See Also

GetFindData() method.

OnStreamInfo Event

Purpose

Periodically notify to the client the date and hour of the last image received and visualized.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

OnStreamInfo(long window, long* date, long status)

Input parameters

Type	Name	Description
long	window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long*	date	Pointer to an _int64 containing the date and hour of the last image visualized in the window specified in the field "window".
long	status	Unused

Remarks

The period of this event can be modified or disabled by calling the RequestStreamInfo method.

See also

RequestStreamInfo() method.

OnAuxStatus Event

Purpose

Return the status of the AUXes.

Compatibility

Proxima: no
Linearis: yes
Spectiva: yes

Syntax

OnAuxStatus(long * _status, long _size, long _connection)

Input parameters

Type	Name	Description
long*	_status	Pointer to a ($_size/4$) DWORD array containing a bit-field of infos about the AUXes availability. See remarks for details.
long	_size	Size in bytes of the array pointed at _status
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Remarks

As this event supports virtually infinite AUXes it is necessary to know the size of the data returned. This size is present in the _size parameter and is expressed in bytes, so the number of DWORD returned is $_size/4$. The number of DWORD returned is given by the number of AUXes present, divided by 32 then rounded up.

The n DWORD array is build up this way:

DWORD 0: 0..31 = AUXes 0 .. 31

DWORD 1: 0..31 = AUXes 32 .. 63

...

DWORD N: 0..31 = AUXes $N*32$.. $(N+1)*32-1$

OnAlarmStatus Event

Purpose

Return the status of the alarms.

Compatibility

Proxima: since release 3.05.00

Linearis: yes

Spectiva: yes

Syntax

OnAlarmStatus (long * _status, long _size, long _connection)

Input parameters

Type	Name	Description
long*	_status	Pointer to a (_size/4) DWORD array containing a bit-field of infos about the Alarms availability. See remarks for details.
long	_size	Size in bytes of the array pointed at _status
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Remarks

As this event supports virtually infinite alarms it is necessary to know the size of the data returned. This size is present in the _size parameter and is expressed in bytes, so the number of DWORD returned is _size/4. The number of DWORD returned is given by the number of alarms present, divided by 32 then rounded up.

The n DWORD array is build up this way:

DWORD 0: 0..31 = alarms 0 .. 31

DWORD 1: 0..31 = alarms 32 .. 63

...

DWORD N: 0..31 = alarms N*32 .. (N+1)*32-1

1 stands for "turned on", 0 for "turned off".

OnAuxAvailableEx Event

Purpose

Return the availability of the AUXes; USB extension board is supported.

Compatibility

Proxima: since release 3.05.00

Linearis: yes

Spectiva: yes

Syntax

OnAuxAvailableEx(long * _status, long _size, long _connection)

Input parameters

Type	Name	Description
long*	_status	Pointer to bitfield of _size bytes, organized into DWORD (32 bits). So _status[0] contains availability of AUXes from 0 to 31, _status[1] contains availability of AUXes from 32 to 63, and so on. 1 stands for "available", 0 for "not available".
long	_size	Size in bytes of the data structure pointed at _status.
long	_connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Remarks

The 4 DWORD array is build up this way:

DWORD 0: 0..31 = AUXes 0..31

DWORD 1: 0..31 = AUXes 32..63

DWORD 2: 0..31 = AUXes 64..95

DWORD 3: 0..31 = AUXes 96..127

Bit set (1) means state is ON.

Bit clear (0) means state is OFF.

OnChannelsAudioAvailable Event

Purpose

Notify that the ActiveX has received a new setup file from the server, and specify which audio channels are available.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: no

Syntax

OnChannelsAudioAvailable(long _status, long _connection)

Input parameters

Type	Name	Description
long	status	<p>A 32 bit bit-field.</p> <p>Proxima: Only the 16 less significant bits are used. Less significant bit is channel 1, the most significant is channel 16. 1 stands for "this channel exists". 0 stands for "this channel does NOT exists".</p> <p>Spectiva: All the 32 bits are used. Less significant bit is channel 1, the most significant is channel 32. 1 stands for "this channel exists". 0 stands for "this channel does NOT exists".</p>
long	_connection	<p>ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.</p>

See also

AudioChannelsOn event.



OnSectorsAvailable Event

Purpose

Specify which sectors are available for playback for each camera.

Compatibility

Proxima: yes

Spectiva: yes

Syntax

OnSectorsAvailable (long * _status, long _connection)

Input parameters

Type	Name	Description
long	<i>_status</i>	A pointer to a cameraSectorStatus. See remarks for details.
long	<i>_connection</i>	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Remarks

Every time an image is received this data are checked. As a difference occurs, this event is launched.

```
struct cameraSectorStatus
{
    DWORD Prime;
    DWORD TLapse;
    DWORD Alarm;
};
```

Each DWORD contains playback availability for each camera.

Bit [0..31] stand for camera [1..32]

Values:

0: sector unavailable for playback;

1: sector available for playback.

OnWinStatusChange Event

Purpose

Notify to the client that it is not possible to continue playback of the actual sector, so playback will be switched to another sector.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

```
OnWinStatusChange(long _window, long _status, long _reason)
```

Input parameters

Type	Name	Description
long	window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long	_status	The sector to which playback will switch to. See Remarks for details.
long	_reason	Error code of the error occurred. See Remarks for details.

Remarks

_status Constants:

```
STATUS_LIVE           = 0 // Switching to Live.
STATUS_PLAY           = 1 // Switching to Playback.
STATUS_PRIME          = 2 // Switching to Prime sector.
STATUS_TLAPSE         = 3 // Switching to Time Lapse sector.
STATUS_ALARM          = 4 // Switching to Alarm sector.
MASTER_WIN            = 5 // [Spectiva only] Master camera in multiple
                          // playback removed.
```

_reason

Constants:

```
NODATA_AVAIL         = 0 // No more playback data available.
PLAY_MULT              = 1 // [Spectiva only] Switching to multiple playback.
```



PLAY_MULT_STOPPED = 2 // [Spectiva only] Multiple playback stopped.
MASTER_REMOVED = 3 // [Spectiva only] Master camera playback windows
// closed while in multiple playback.



OnExportData Event

Purpose

As a consequence of a call to the ExportStart() method, this event is triggered when block of data are sent.

Compatibility

Proxima: no
Linearis: yes
Spectiva: yes

Syntax

OnExportData(long connection, long streamId, long* buffer, long bufferSize)

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	streamId	A valid stream ID returned by ExportStart() method.
long*	buffer	Pointer to a buffer containing data to be processed to export purpose
long	bufferSize	Size of the buffer pointed.

OnExportStatus Event

Purpose

Notify the status of the export process.

Compatibility

Proxima: no

Linearis: yes

Spectiva: yes

Syntax

OnExportStatus(long connection, long streamId, long status)

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	streamId	A valid stream ID returned by ExportStart() method.
long	status	Status of the export process: 18 = Export aborted 19 = Export finished 76 = Export failed - user has no permission - a smart-card certificate is needed

OnCameraAuxAvailable Event

Purpose

Notify to the client the availability of AUX connectors belonging to each camera.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: no

Syntax

OnCameraAuxAvailable(long * status, long size, long connection)

Input parameters

Type	Name	Description
long*	status	An array of size/4 DWORD, containing in each DWORD the availability of AUXes belonging to each camera. See Remarks for details.
long	size	Size in bytes of the data structure pointed at *status.
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Remarks

Up to 32 AUXes are supported for each camera.

DWORD 0 contains AUX availability for camera 0, DWORD 1 for camera 1 and so on...

In each DWORD:

bit 0: 1 stands for AUX 0 available, 0 for AUX 0 not available;

...

bit 31: 1 stands for AUX 31 available, 0 for AUX 31 not available.

OnCameraAuxStatus Event

Purpose

Notify to the client the status of AUX connectors belonging to each camera.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: no

Syntax

OnCameraAuxStatus(long * status, long size, long connection)

Input parameters

Type	Name	Description
long*	status	An array of size/4 DWORD, containing in each DWORD the status of AUXes belonging to each camera. See Remarks for details.
long	size	Size in bytes of the data structure pointed at *status.
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

Remarks

DWORD 0 contains AUX status for camera 0, DWORD 1 for camera 1 and so on...

In each DWORD:

bit 0: 1 stands for AUX 0 closed, 0 for AUX 0 open;

...

bit 31: 1 stands for AUX 31 closed, 0 for AUX 31 open.

OnCameraAux Event

Purpose

Notify to the client the status of the specified AUX connector belonging to the specified camera.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: no

Syntax

OnCameraAux(long camera, long aux, long status, long connection)

Input parameters

Type	Name	Description
long	camera	Number of the camera whose aux has been modified.
long	aux	Number of the modified aux.
long	status	New status of the modified aux: 1 means closed; 0 means open.
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

OnRecordingMode Event

Purpose

Notify to the client the recording status of the specified camera on the specified camera.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: no

Syntax

OnRecordingMode(long camera, long sector, long manualRec, long genericRec, long connection)

Input parameters

Type	Name	Description
long	camera	Number of the camera for which details are given.
long	sector	Number of the sector for which details are given.
long	manualRec	Real mode of the recording: 0 automatic (normal state of after a "ResetManualRec" method call) 1 manual recording (after a "StartManualRec" method call) 2 manual stop (after a "StopManualRec" method call)
long	GenericRec	Boolean: 1 if the sector is affectively in recording state, 0 otherwise.
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.

OnAlarmEx Event

Purpose

Returns extended informations on a triggered alarm.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: yes

Syntax

OnAlarmEx (long connection, long alarmId, long alarmType, BSTR alarmDescription, BSTR conditionsDescription, BSTR conditionsStatus, long alarmStatus, long* serverDate)

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	alarmId	Number of the alarm.
long	AlarmType	Type of triggered alarm. See remarks for details.(<u>Spectiva only</u>)
BSTR	AlarmDescription	Textual description of the alarm.
BSTR	conditionsDescription	Full textual description of a custom condition. (<u>Spectiva only</u>)
BSTR	conditionsStatus	Logical status of each condition of the logical predicate of the custom condition.(<u>Spectiva only</u>)
long	alarmStatus	Boolean: 1 for alarm active, 0 for alarm not active.
long*	serverDate	Pointer to an _int64 containing the server hour and date at which the alarm was triggered.

Remarks

AlarmType Constants values (Spectiva only)

ALARMGROUP_NONE	-1
ALARMGROUP_MOTION	0
ALARMGROUP_VIDEOLOSS	1
ALARMGROUP_SWITCH	2



OnNeedCertificate Event

Purpose

Passes to the client an encrypted buffer to be decrypted and sent back.

Compatibility

Proxima: no

Linearis: yes

Spectiva: yes

Syntax

OnNeedCertificate(long * buffer, long size, long connection)

Input parameters

Type	Name	Description
long*	buffer	A buffer containing encrypted data to be decrypted and sent back to the server as user validation
long	size	Size in bytes of the data structure pointed at *buffer.
long	connection	ID of server connection.

Remarks

buffer contains encrypted data. This data has to be decrypted, using the appropriate certificate, then this decrypted data has to be sent back to the server calling the SendCertificate method to continue the login process.

See also

SendCertificate method.

OnNeedCertificateBuffer Event

Purpose

Passes to the client an encrypted buffer to be decrypted and sent back using the certificate whose serial is given.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

OnNeedCertificateBuffer(long* serial, long serialSize, long* buffer, long size, long connection)

Input parameters

Type	Name	Description
long*	serial	A data buffer containing the certificate serial.
long	serialSize	Size in bytes of the certificate serial data buffer.
long*	buffer	A buffer containing encrypted data to be decrypted and sent back to the server as user validation
long	size	Size in bytes of the data structure pointed at *buffer.
long	connection	ID of server connection.

Remarks

buffer contains encrypted data. This data has to be decrypted, using the appropriate certificate, then this decrypted data has to be sent back to the server calling the SendCertificateBuffer method to continue the login process.

The certificate to use to decrypt the given buffer is the one whose serial is specified by the serial field,

See also

SendCertificateBuffer method.

OnNeedCertificatePlay Event

Purpose

Passes to the client an encrypted buffer to be decrypted and sent back to display some encrypted video footage.

Compatibility

Proxima: no
Linearis: yes
Spectiva: no

Syntax

OnNeedCertificatePlay(long* buffer, long size, long connection)

Input parameters

Type	Name	Description
long*	buffer	A buffer containing encrypted data to be decrypted and sent back to the server as user validation
long	size	Size in bytes of the data structure pointed at *buffer.
long	connection	ID of server connection.

Remarks

buffer contains encrypted data. This data has to be decrypted, using the appropriate certificate, then this decrypted data has to be sent back to the server calling the SendCertificate method to continue the playback process.

See also

SendCertificate method.

OnSetPriorityCamera Event

Purpose

Notify the client when a priority camera is set.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

OnSetPriorityCamera(long connection, long camera)

Input parameters

Type	Name	Description
long	connection	ID of server connection.
long	camera	Number of the camera with priority.

Remarks

camera can be -1 if no priority is set..

See also

CanSetPriorityCamera method.

SetPriorityCamera method.

OnAudioChannelsOnEx Event

Purpose

Notify the client which audio channels is active on the given camera.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

OnAudioChannelsOnEx(long connection, long* channels, long camera)

Input parameters

Type	Name	Description
long	connection	ID of server connection.
long*	channels	Pointer to a _int64 bitfield containing the availability of each single audio channels.
long	camera	Number of the camera.

Remarks

In the channels variable, when the bit at position 0 is set to 1 means that the first audio channel is enabled, when the bit at position 1 is set to 1 means that the first audio channel is enabled and so on...

See also

OnAudioChannelsAvailableEx event.

OnChannelsAudioAvailableEx Event

Purpose

Notify the client which audio channels is associated to the given camera.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

OnChannelsAudioAvailableEx(long connection, long* channels, long camera)

Input parameters

Type	Name	Description
long	connection	ID of server connection.
long*	channels	Pointer to a _int64 bitfield containing the availability of each single audio channels.
long	camera	Number of the camera.

Remarks

In the channels variable, when the bit at position 0 is set to 1 means that the first audio channel is available, when the bit at position 1 is set to 1 means that the first audio channel is available and so on...

See also

OnAudioChannelsOnEx event.

OnExportTIData Event

Purpose

This event is triggered during the export process, to pass any text insertion data if present within the recordings. This data can be directly processed by the user, or passed as is to the archiveexport ActiveX control, or simply dropped,

Compatibility

Proxima: no
Linearis: yes
Spectiva: yes

Syntax

OnExportTIData(long connection, long StreamID, long* buffer, long buffersize)

Input parameters

Type	Name	Description
long	connection	ID of server connection.
long	StreamID	The Stream ID returned by ExportStart() method.
long*	buffer	A raw data buffer containing Text insertion data
long	bufferSize	The size (in bytes) of the buffer.

OnClick Event

Purpose

This event is triggered when the left mouse button is clicked within a window belonging to the ActiveX Control.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

OnClick(long window, long X, long Y)

Input parameters

Type	Name	Description
long	window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
long	X	Abscissa of the clicked point referred to the window. The origin is located in the upper left corner.
long	Y	Ordinate of the clicked point referred to the window. The origin is located in the upper left corner.

OnAudioChannelsOn Event

Purpose

Notify the client which audio channels is active for the given camera.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

OnAudioChannelsOn(long connection, long camera, VARIANT channels)

Input parameters

Type	Name	Description
long	connection	ID of server connection.
long	camera	Number of the camera.
VARIANT	channels	A VARIANT structure containing a boolean vector. The variant itself describes the data structure.

Remarks

See the ChannelsAudioOn event to manage the audio channels on Linearis and/or Proxima DVMSes.

AudioChannelsOn Event

Purpose

Notify the client which audio channels is active for the given camera.

Compatibility

Proxima: yes

Linearis: yes

Spectiva: no

Syntax

AudioChannelsOn(long connection, long status, long camera)

Input parameters

Type	Name	Description
long	connection	ID of server connection.
long	status	A bitfield containing the availability of each single audio channel for the given camera. See remarks for details.
long	camera	Number of the camera.

Remarks

See the OnChannelsAudioOn event to manage the audio channels on a Spectiva DVMS.

In the status variable, when the bit at position 0 is set to 1 means that the first audio channel is available, when the bit at position 1 is set to 1 means that the first audio channel is available and so on...

OnAudioChannelsAvailable Event

Purpose

Notify the client which audio channels is available.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

OnAudioChannelsAvailable(long connection, VARIANT channels)

Input parameters

Type	Name	Description
long	connection	ID of server connection.
VARIANT	channels	A VARIANT structure containing a boolean vector. The variant itself describes the data structure.

OnAlarmsAvailable2 Event

Purpose

Return the availability of alarms connector, both physical or logical.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

OnAlarmsAvailable2(long connection, VARIANT status)

Input parameters

Type	Name	Description
long	connection	ID of server connection.
VARIANT	status	A VARIANT structure containing a boolean vector. The variant itself describes the data structure.

OnAlarmStatus2 Event

Purpose

Return the status of up to 128 alarms. USB extension board IS supported.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

OnAlarmStatus2(long connection, VARIANT status)

Input parameters

Type	Name	Description
long	connection	ID of server connection.
VARIANT	status	A VARIANT structure containing a boolean vector. The variant itself describes the data structure.

OnAlarmEx2 Event

Purpose

Returns extended informations on a triggered alarm.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

OnAlarmEx (long connection, long alarmId, long alarmType, BSTR alarmDescription, BSTR conditionsDescription, BSTR conditionsStatus, long alarmStatus, long* serverDate)

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
long	alarmId	Number of the alarm.
long	AlarmType	Type of triggered alarm. See remarks for details.
BSTR	AlarmDescription	Textual description of the alarm.
BSTR	conditionsDescription	Full textual description of a custom condition. (<u>Spectiva only</u>)
BSTR	conditionsStatus	Logical status of each condition of the logical predicate of the custom condition. (<u>Spectiva only</u>)
long	alarmStatus	Boolean: 1 for alarm active, 0 for alarm not active.
DATE	serverDate	The server hour and date at which the alarm was triggered.

Remarks

AlarmType Constants values

ALARMGROUP_NONE	-1
ALARMGROUP_MOTION	0
ALARMGROUP_VIDEOLOSS	1
ALARMGROUP_SWITCH	2
ALARMGROUP_IMAGEBEHAVIOUR	3

OnAuxStatus2 Event

Purpose

Return the status of up to 128 auxes. USB extension board IS supported.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

OnAuxStatus2(long connection, VARIANT status)

Input parameters

Type	Name	Description
long	connection	ID of server connection.
VARIANT	status	A VARIANT structure containing a boolean vector. The variant itself describes the data structure.

OnAuxAvailableEx2 Event

Purpose

Return the availability of up to 128 AUX connector. USB extension board IS supported.

Compatibility

Proxima: no
Linearis: no
Spectiva: yes

Syntax

OnAuxAvailableEx2(long connection, VARIANT status)

Input parameters

Type	Name	Description
long	connection	ID of server connection.
VARIANT	status	A VARIANT structure containing a boolean vector. The variant itself describes the data structure.

OnCameraAuxAvailable2 Event

Purpose

Return available AUXes for each camera.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

OnCameraAuxAvailable2(long connection, VARIANT status)

Input parameters

Type	Name	Description
long	connection	ID of server connection.
VARIANT	status	A VARIANT structure containing a boolean vector. The variant itself describes the data structure.

OnCameraAuxStatus2 Event

Purpose

Return the status of the AUXes of each camera.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

OnCameraAuxStatus2(long connection, VARIANT status)

Input parameters

Type	Name	Description
long	connection	ID of server connection.
VARIANT	status	A VARIANT structure containing a boolean vector. The variant itself describes the data structure.

OnStreamInfo2 Event

Purpose

Periodically notify to the client the date and hour of the last image received and visualized.

Compatibility

Proxima: no
Linearis: no
Spectiva: yes

Syntax

OnStreamInfo2(long window, DATE date, long status)

Input parameters

Type	Name	Description
long	window	ID of a visualization window. This must be a valid window ID returned by SetExternalWin() or CreateWin() methods.
DATE	date	The date and hour of the last image visualized in the window specified in the field "window".

Remarks

The period of this event can be modified or disabled by calling the RequestStreamInfo method.

See also

RequestStreamInfo method.

OnSectorsAvailable2 Event

Purpose

Specify which sectors are available for playback for each camera.

Compatibility

Proxima: no

Linearis: no

Spectiva: yes

Syntax

OnSectorsAvailable2(long connection, VARIANT prime, VARIANT tlapse, VARIANT alarm)

Input parameters

Type	Name	Description
long	connection	ID of server connection. This must be a valid server connection ID returned by Connect() or CryptConnect() method.
VARIANT	prime	A VARIANT structure containing a boolean vector. The variant itself describes the data structure.
VARIANT	tlapse	A VARIANT structure containing a boolean vector. The variant itself describes the data structure.
VARIANT	alarm	A VARIANT structure containing a boolean vector. The variant itself describes the data structure.

Remarks

Every time an image is received this data are checked. As a difference occurs, this event is launched.