



Introducing

The Milestone Integration Platform

Software Development Kit 1.0

Milestone Systems - Confidential



The Open Platform Company

Agenda



- ◆ Introduction
- ◆ Product Overview
- ◆ Technical Introduction
- ◆ Summary
- ◆ Q&A





Product Overview

Jan Lindeberg



Agenda

- ◆ Milestone Integration Platform overview
- ◆ Milestone Solution Partner – MSP integration possibilities
- ◆ What makes MIP different from the present SDK?
- ◆ Availability and rollout plan



Milestone Integration Platform

Introduction

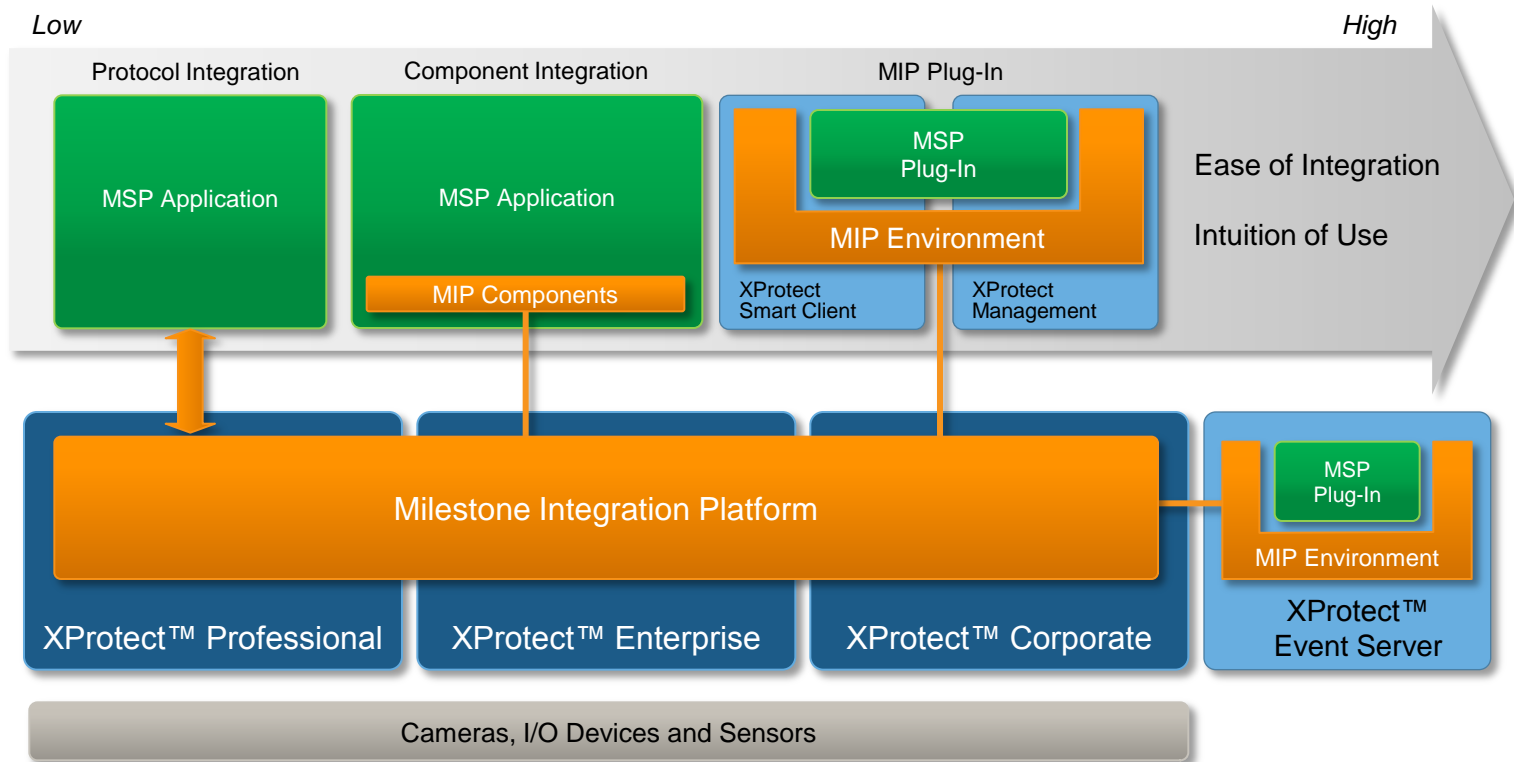


- ◆ Next generation SDK
- ◆ Versatile integration possibilities:
 - ◆ Verticalization
 - ◆ Customization
- ◆ Cornerstone in Milestone Open Platform strategy
- ◆ Foundation for the 300+ Milestone Solution Partners eco-system



Milestone Integration Platform

Versatile integration possibilities



What 's New in MIP-SDK?



- ◆ Application plug-in capability via the MIP environment
 - ◆ Management plug-in
 - ◆ Extended Smart Client plug-in with configuration possibilities
 - ◆ Server side plug-in
- ◆ Full compatibility
 - ◆ XProtect product and version agnostic
 - ◆ MIP SDK forward compatible
- ◆ Extended SDK capabilities
- ◆ Comprehensive development toolbox



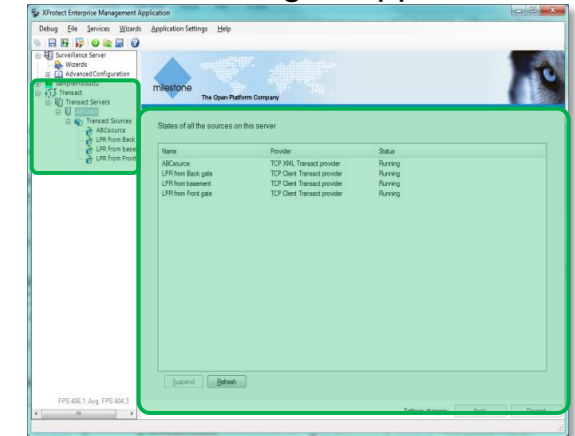
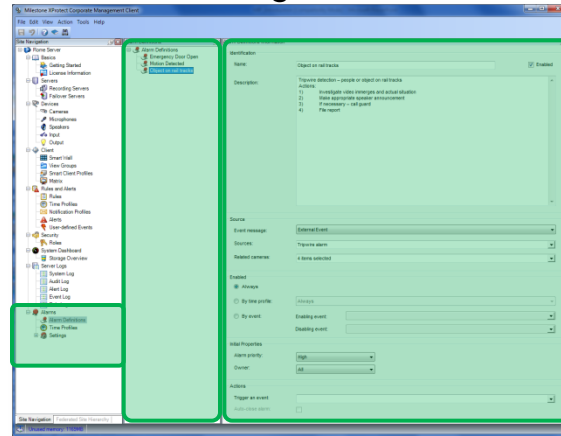
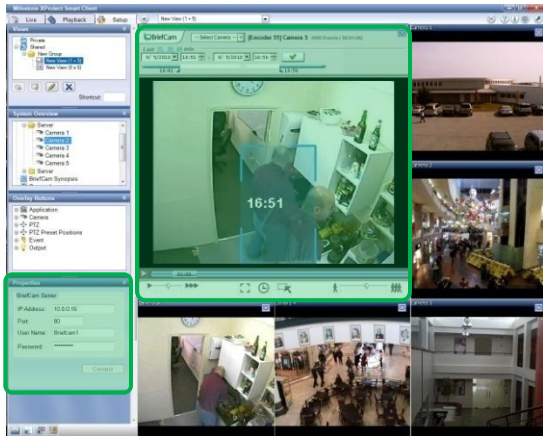
MIP Plug-In – Intuitive Integration

Seamless user experience



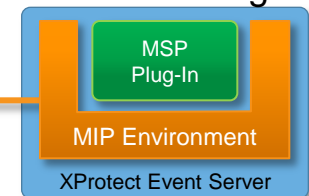
Share MIP plug-ins between XProtect products

- 1 XProtect Smart Client
- 2 XPCO Mgmt Client
- 3 XPE/XPP Mgmt Application



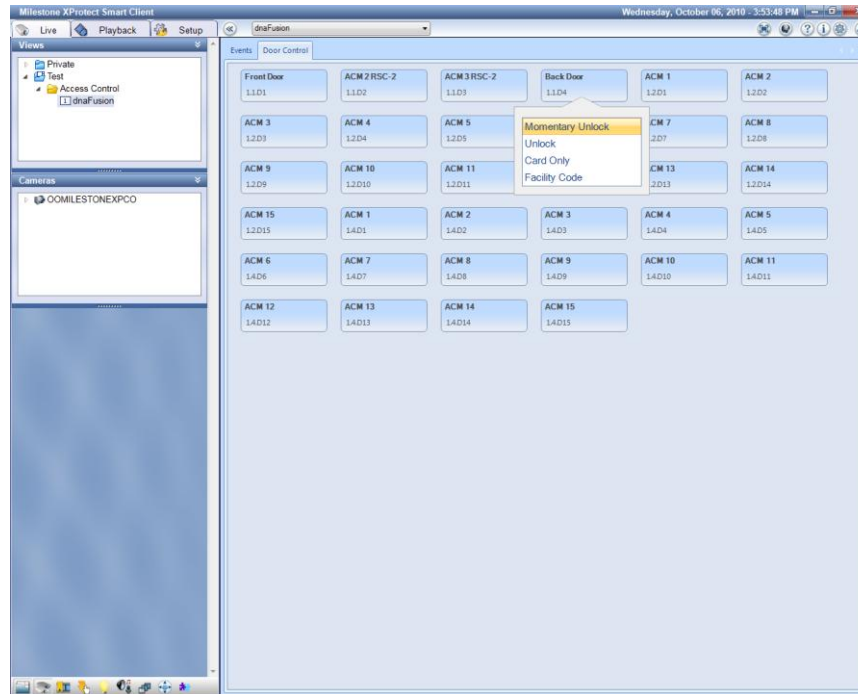
Milestone Integration Platform

- 4 Server side Plug-Ins



MIP Plug-In Example

Open Options Inc. – Access Control

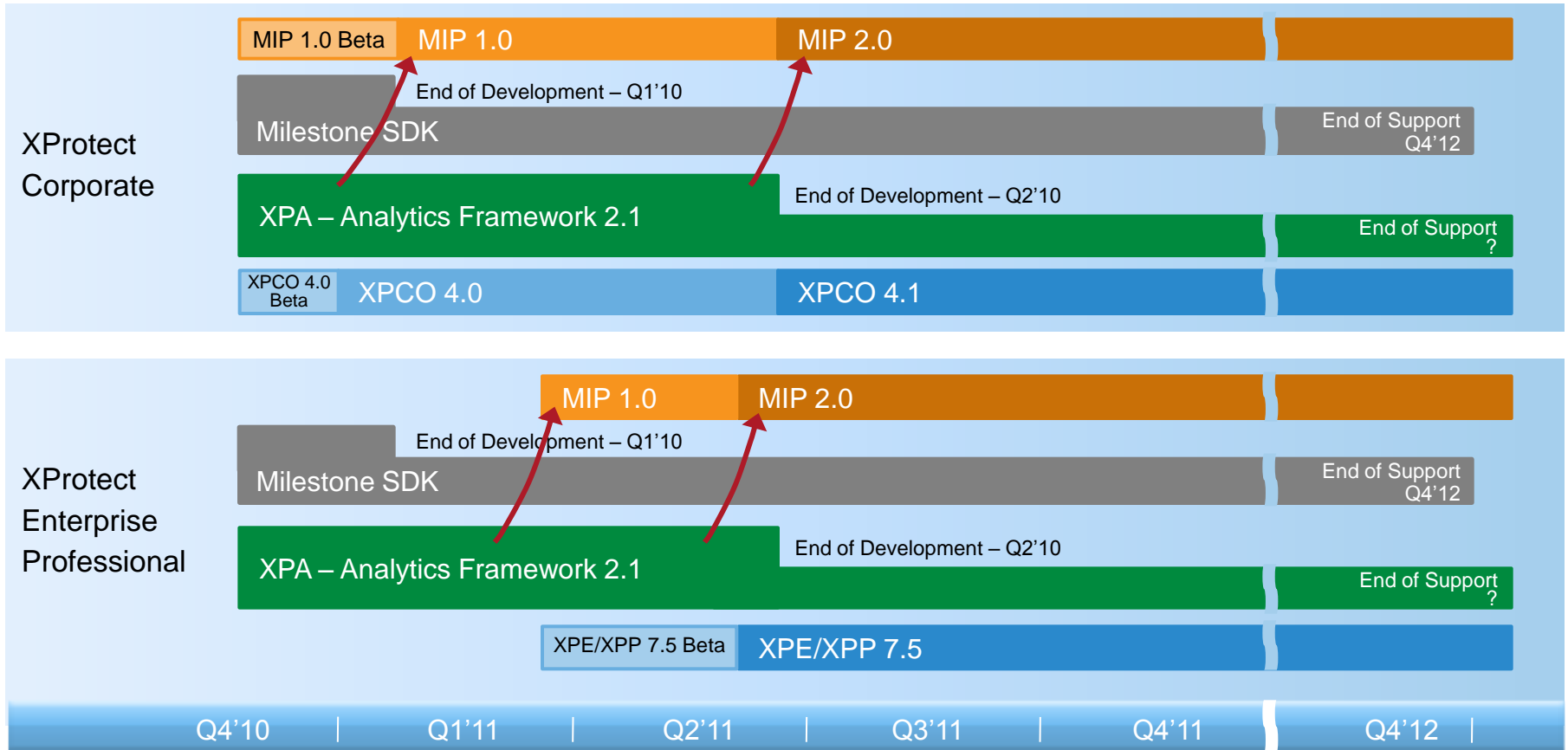


“Working with the new Milestone MIP SDK has been a breeze. I found the SDK layout to be very intuitive and well thought out. This SDK will allow us to deliver an even tighter integrated solution with lots of new features to our customers in a very short time frame.”

- George Crawford, Software Developer, Open Options Inc.



MIP SDK Rollout Plan





Technical Introduction

Anders Bent Christensen



milestone

Agenda

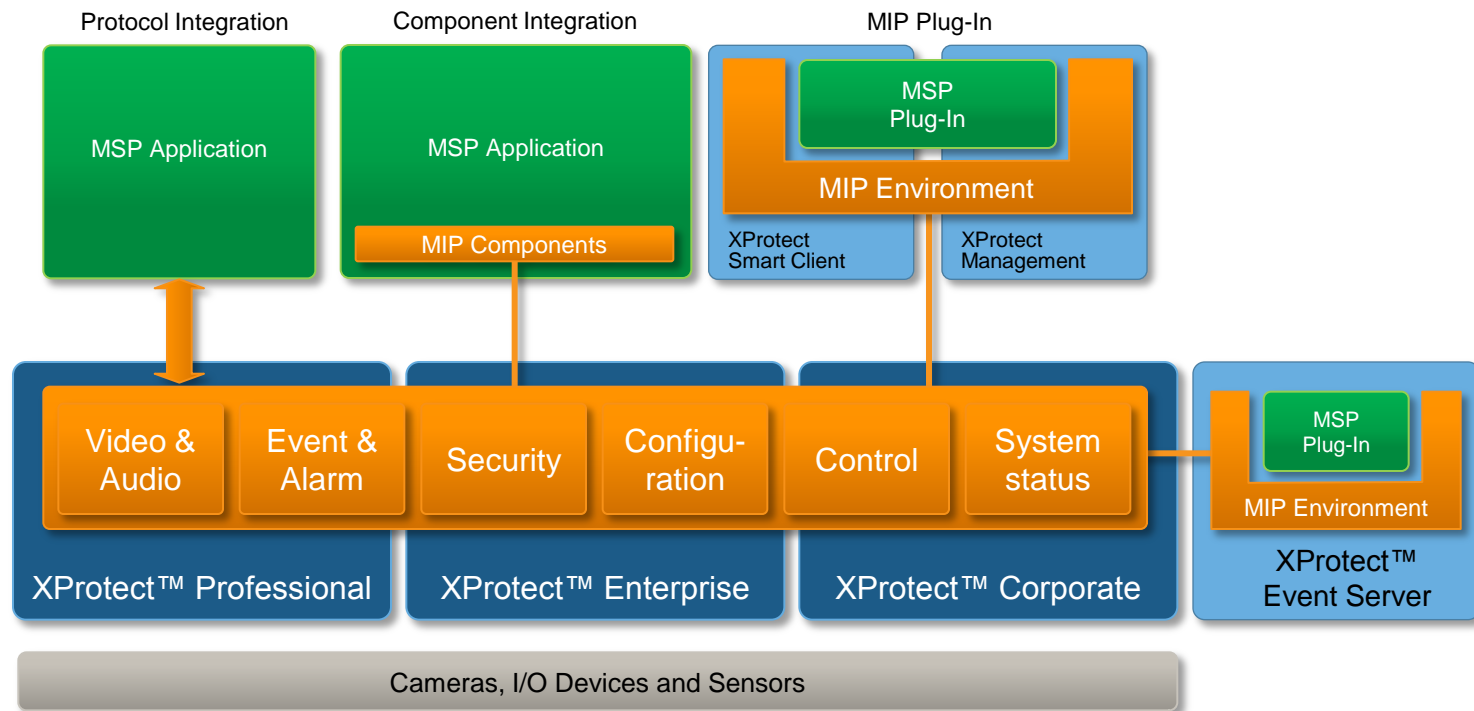


- ◆ MIP SDK Feature Overview
- ◆ Architecture overview
- ◆ Applications & samples
 - ◆ Access Control
 - ◆ Analytics Overlay on Video
- ◆ Methodology change
- ◆ MIP SDK content
- ◆ A programming sample – video overlay



Milestone Integration Platform

Feature groups



MIP SDK 1.0 Features

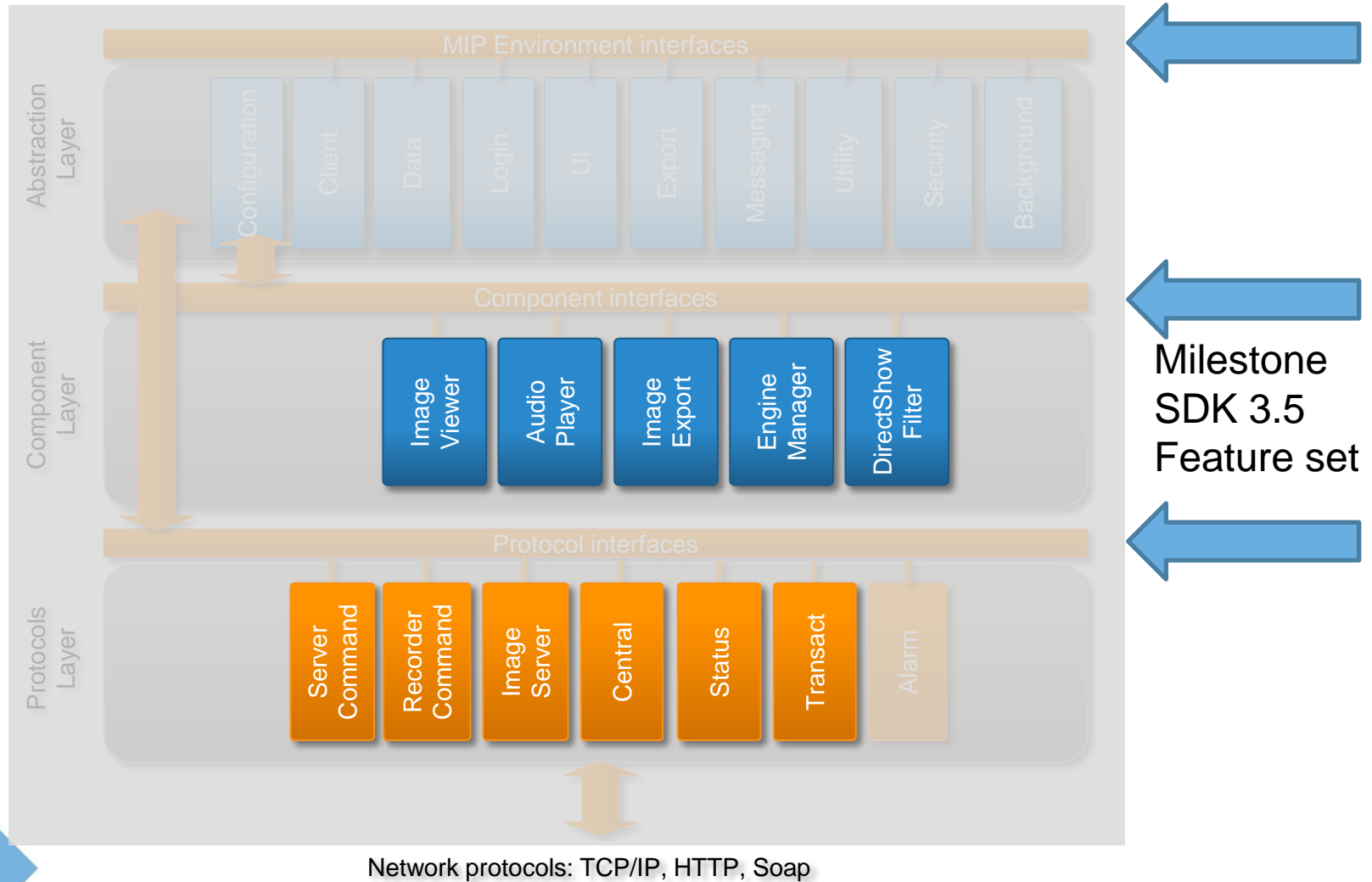


- ◆ Video & Audio
 - ◆ Retrieve live/recorded video & audio
 - ◆ Retrieve JPEG
 - ◆ Manage export component
 - ◆ Place overlay on live / recorded video
- ◆ Event & Alarm
 - ◆ Trigger events (internal or external)
 - ◆ Create new alarms
 - ◆ Retrieve existing list of events
- ◆ Security
 - ◆ Login assistance
 - ◆ Plug-in authorization
- ◆ Configuration
 - ◆ Retrieve MVS configuration
 - ◆ Load/Store plug-in configuration
- ◆ Control
 - ◆ Commands to PTZ cameras
 - ◆ Start / Stop Recording
 - ◆ Activate output (AUX)
 - ◆ Smart Wall and Matrix control
 - ◆ Commands to other plug-ins
- ◆ System Status
 - ◆ Central protocol
 - ◆ Status protocol



Milestone Integration Platform

Architecture Overview



Feature / Layer Matrix

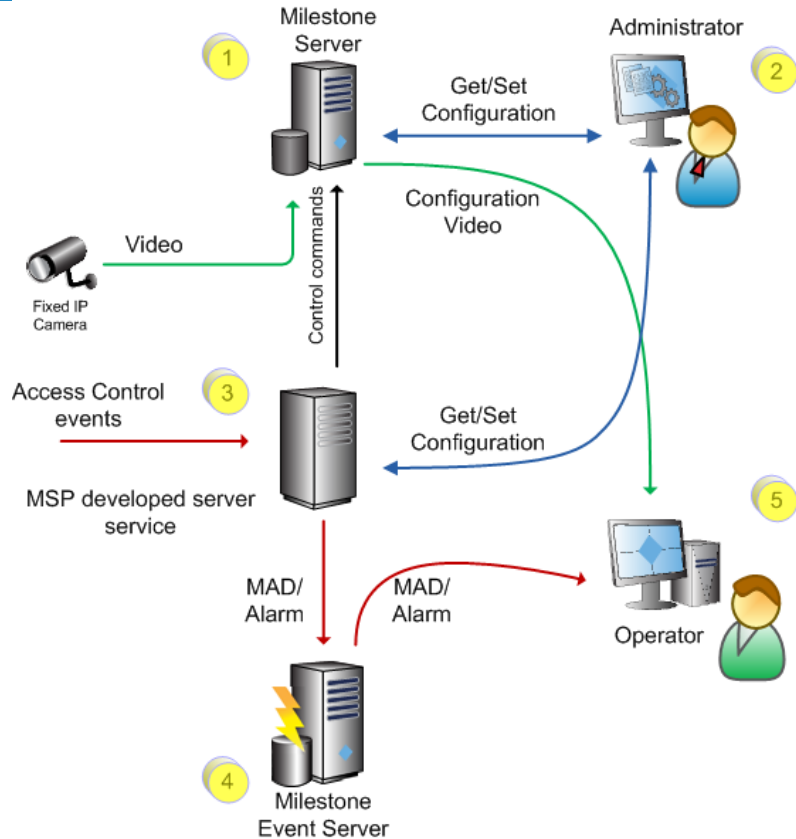


Layer	Function	Video & Audio	Event & Alarm	Security	Configuration	Control	System Status
MIP Abstraction Layer	Configuration				✓		
	Client	✓	✓		✓	✓	
	Data		✓				
	Login			✓			
	UI	✓			✓		
	Export	✓					
	Messaging		✓	✓	✓	✓	
	Utility			✓	✓		
	Security			✓			
	Background					✓	✓
Component Layer	Image Viewer	✓				✓	
	Audio Player	✓					
	Image export	✓					
	Engine Manager			✓	✓		
	DirectShow filter	✓					
Protocol Layer	Server Command			✓	✓	✓	
	Recorder Command	✓	✓	✓		✓	
	Image Server	✓	✓	✓	✓	✓	
	Central		✓		✓		✓
	Status		✓				✓
	Transact		✓			✓	
	Alarm		✓				

milestone

Access Control

Applications & Samples

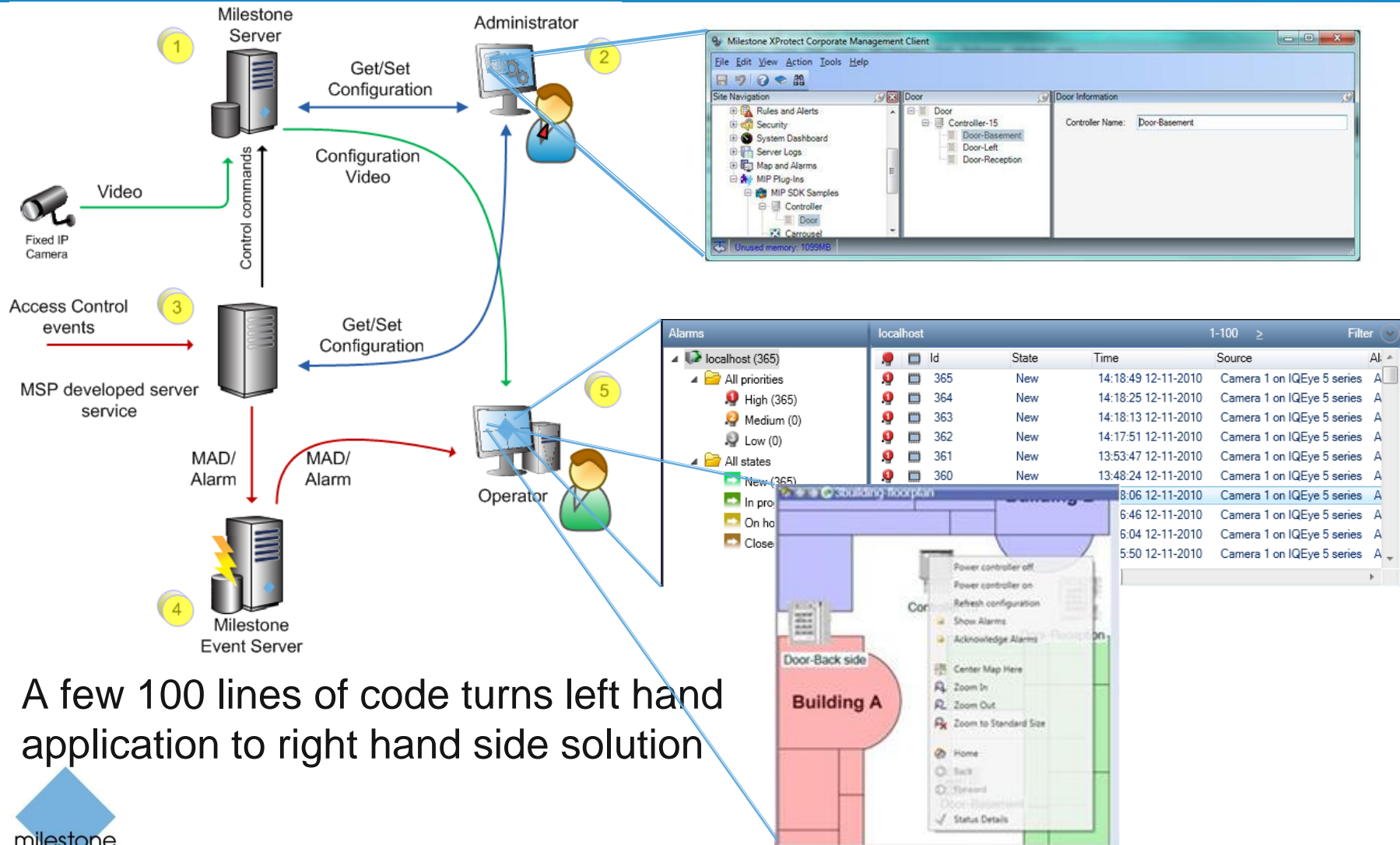


- ◆ Development at (2)
 - ◆ Configure Sensor to Camera relations
 - ◆ Send configuration to MSP server
- ◆ Development at (3)
 - ◆ Send event in MAD format
 - ◆ Retrieve configuration
 - ◆ Sensor / camera relationship
 - ◆ IP address for Event Server



Access Control

Applications & Samples

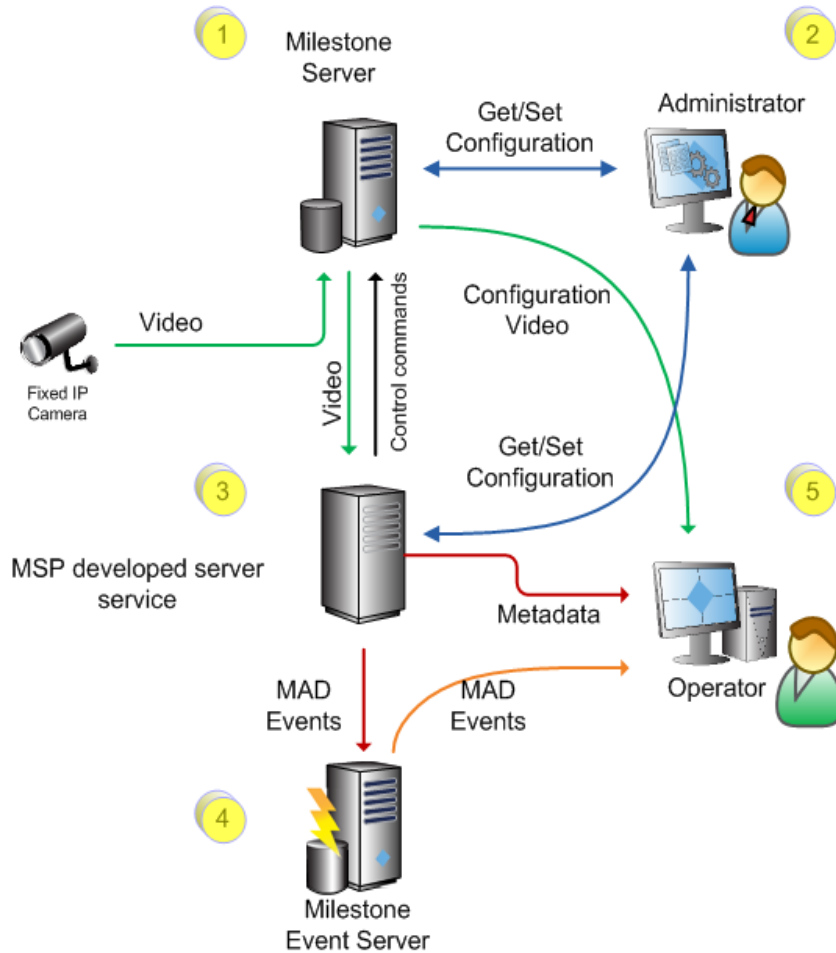


- ◆ A few 100 lines of code turns left hand application to right hand side solution



Smart Client video analytics overlay

Applications & Samples

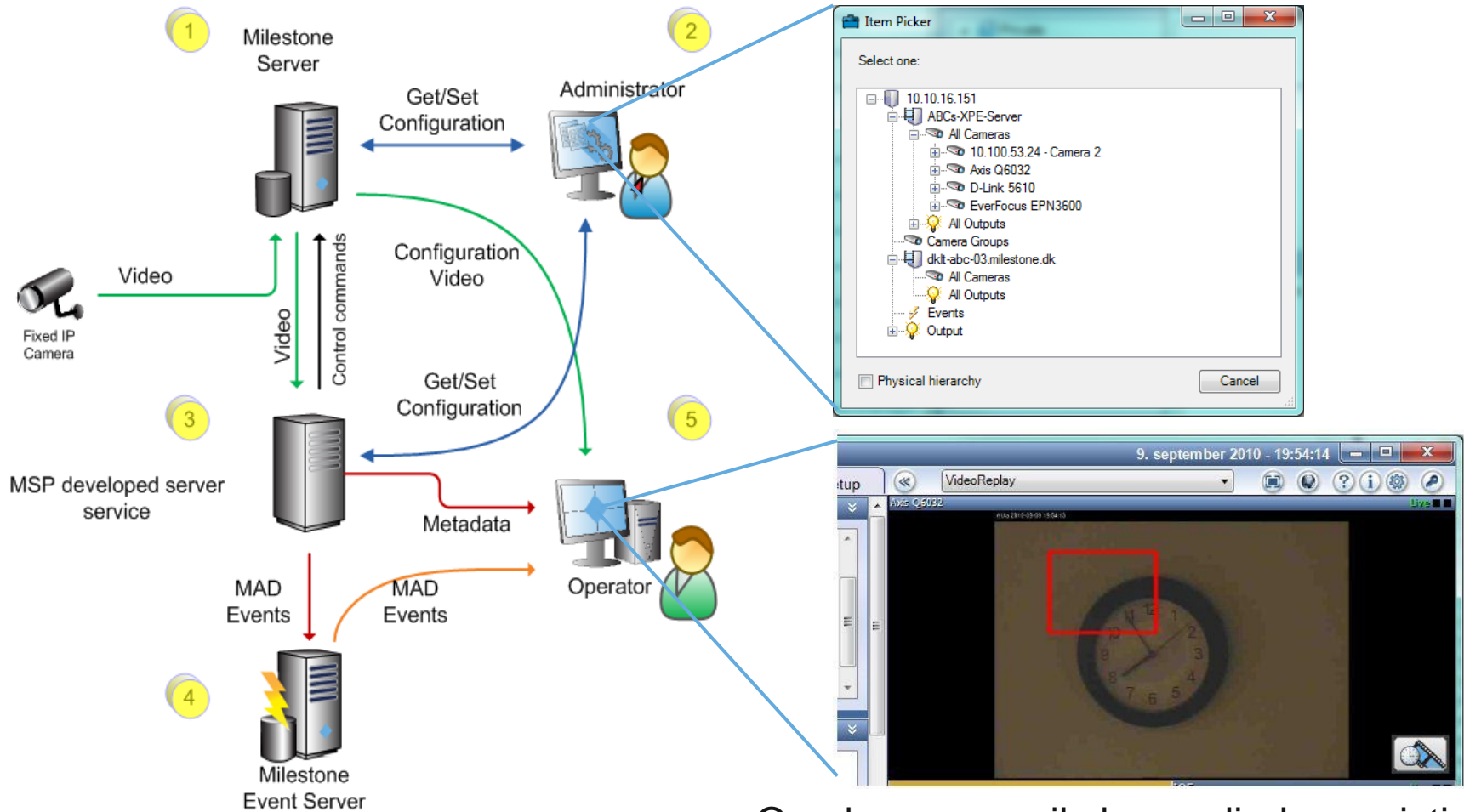


- ◆ Development at (2)
 - ◆ Configure analytics / camera relationship
- ◆ Development at (3)
 - ◆ Send event in MAD format
 - ◆ Store analytics metadata indexed by camera and time
 - ◆ Be able to deliver analytics metadata
- ◆ Development at (5)
 - ◆ Retrieve analytics metadata from (3)
 - ◆ Place analytics metadata as overlay on video



Smart Client video analytics overlay

Applications & Samples



- ◆ Overlay can easily be applied on existing CameraViewItem and standalone ActiveX



milestone

Methodology Change

Control of activation – a sample



◆ The MIP SDK way to activate

```
EnvironmentManager.Instance.SendMessage(  
    new Message(_messageId), _item.FQID);
```

Type of Command

Item to perform the command on

```
MessageId.Control.TriggerCommand  
MessageId.Control.StartRecordingCommand  
MessageId.Control.StopRecordingCommand
```

Typical Milestone built-in Item types:

- Camera
- Outputs (Dry contacts)
- AUX
- Presets
- PTZ
- User defined event



Methodology Change

Unification



- ◆ Control and commands unified
 - ◆ Activated same way across different Item types
 - ◆ Camera, Output, Event, Matrix, other Plug-ins
 - ◆ Allow end-user to chose what command to trigger on what item
- ◆ Configuration Items unified
 - ◆ Identification and usage of items inherited from same class
 - ◆ User selection and stored identification works for all item types
- ◆ Result:
 - ◆ Unified commands can operate on unified Items
 - ◆ Same support for
 - ◆ Smart Client plug-in
 - ◆ Event Server plug-in
 - ◆ MIP .Net Library for standalone applications



MIP SDK content

Documentation

- ◆ One UI contains all documentation
- ◆ Search across notes and interface documentation
- ◆ Documentation divided on key development approach
 - ◆ Protocol
 - ◆ Component
 - ◆ Plug-in



The screenshot shows a web browser window titled "MIP SDK 1.0a - Documentation". The page features the Milestone logo and the tagline "The Open Platform Company". A navigation bar includes "Way of Integration:" with sub-links for "Protocol Integration", "Component Integration", and "Plug-in Integration". The main content area is titled "Welcome to the MIP SDK Portal" and includes an "Introduction to Milestone Integration Platform Software Development Kit". Below the text is a diagram illustrating the integration architecture. The diagram shows three integration paths: Protocol Integration (MSP Application), Component Integration (MSP Application and MIP Components), and Plug-in Integration (MSP Plug-in). These paths lead to an "MIP Environment" which includes "XProtect Smart Client" and "XProtect Management". This environment is supported by the "Milestone Integration Platform" and "XProtect™ Servers". At the bottom, "Cameras, I/O Devices and Sensors" are connected to the system. A copyright notice "© Milestone Systems 2010" is visible at the bottom of the page.

MIP SDK content

Documentation – Plug-in Integration



- ◆ Integration approach content
 - ◆ Application catalogue
 - ◆ Architecture notes
 - ◆ Class reference documentation
 - ◆ Getting started
 - ◆ Samples

The screenshot displays the MIP SDK 1.0a - Documentation website. The page is titled "Plug-in Integration" and is part of the "Way of Integration" section. The navigation bar includes "Protocol Integration", "Component Integration", and "Plug-in Integration". The left sidebar contains a table of contents with sections like "Application Catalog", "Architecture", "Reference Documentation", "Getting Started", and "Samples". The main content area is titled "Plug-in Integration" and includes a sub-section "How to Integrate on MIP". This section explains that integration on MIP enables the use of Milestone administration tools, Event Server, and Smart Client for hosting applications, leading to improved ease of use and a common look and feel. It also lists several benefits of the MIP Environment, such as adding system control, storing configurations, sharing configurations, assigning administrator look and feel, adding user controls, and making customized graphics overlay. At the bottom, a diagram illustrates "MIP enabled Applications" with three examples: XProtect Smart Client, XProtect Management, and XProtect Event Server, each containing an "MSP Plug-in" within its "MIP Environment".

Application Catalog

- ▶ Smart Client Enhancement
- ▶ Analytics Integration 1
- ▶ Analytics Integration 2
- ▶ Analytics Integration 3
- ▶ Disk Management
- ▶ Access Control
- ▶ Complete Application Catalog (printable format)

Architecture

- ▶ Introduction to Classes
- ▶ Working with Items
- ▶ Working with Messages
- ▶ Class Interaction Figures
- ▶ Video Display and Overlay
- ▶ Authorization
- ▶ Class Overview and Function
- ▶ Configuration Choices
- ▶ Class Life Cycle

Reference Documentation

- ▶ MIP Reference Manual

Getting Started

- ▶ Plug-in Development
- ▶ Technical Information

Samples

- ▶ Access Control
- ▶ Analytics Overlay
- ▶ Video Preview

Plug-in Integration

How to Integrate on MIP

Integration on MIP enables you to utilize Milestone administration tools, Event Server and Smart Client for hosting your application.

This allows for improved ease of use and help towards a common look and feel for the end user.

A MIP plug-in is able to execute in a number of Milestone products, making it easy to support multiple products with a single developed plug-in.

The MIP Environment allows developers to, for example:

- Add simple ways of system control, e.g. handling devices with PTZ, events and outputs.
- Store MSP developed configurations.
- Share MSP configuration between applications.
- Assign same administrator look and feel for both XProtect Enterprise and XProtect Corporate.
- Add specific user controls to 'Setup', 'Live' and 'Playback' panels inside Smart Client.
- Add customized options menu configuration for common parameter setup, for example; user private and shared between all users.
- Make customized graphics overlay on top of live or recorded video, e.g. for analytics applications.
- Access recorded images for post processing management.

MIP enabled Applications

- XProtect Smart Client
- XProtect Management
- XProtect Event Server

© Milestone Systems 2010



MIP SDK content

Documentation - Search

- ◆ Sample: Search for 'Preset'
- ◆ Left hand side contains result
- ◆ Right hand side contains selected result

The screenshot shows the MIP SDK 1.0a Documentation search results page. The page is titled "MIP SDK 1.0a - Documentation" and features the Milestone logo and "The Open Platform Company" text. The search results are displayed in a list on the left, and the selected result, "Architecture Working with Messages", is shown in detail on the right.

Search Result

- ImageServer - Requests and Responses
- DirectShow Filter
- MIP Environment: VideoOS.Platform.FGID Reference
- MIP Environment: VideoOS.Platform.Kind Reference
- MIP Environment: Member List
- MIP Environment: Class Members
- MIP Environment: Class Members - Variat
- Video Display and Overlay
- ImageServer - Getting the Devices
- Configuration Dump
- MIP Environment: VideoOS.Platform.UI.UTI Reference
- MIP Environment: Member List
- MIP Environment: VideoOS.Platform.Configuration Class Reference
- MIP Environment: VideoOS.Platform.Item Reference
- MIP Environment: VideoOS.Platform.Messaging.MessageId Class Reference
- MIP Environment: Configuration.cs File Reference
- MIP Environment: VideoOS.Platform.Name Reference
- MIP Environment: Class List
- Working with Items
- **Working with Messages**
- EngineManager ActiveX Control
- Server Side Carousel
- MIP Environment: Class Members
- MIP Environment: Class Members - Variat

Architecture

Working with Messages

A message system is used to communicate within one application. The application sends a number of messages that the plug-in can subscribe to, and the plug-in can also send messages.

A plug-in can make its own messages if it needs to communicate with other plug-ins, whereby only the two plug-ins are involved in exactly that communication. But a plug-in can also utilize the large set of built-in messages that controls the application, trigger custom events, outputs, matrix monitor or camera PTZ [presets](#).

A plug-in can trigger actions and events on items that the plug-in does not fully understand; all it needs to look for is that the item is of category: `Category.TriggerOut` item.

All message communication is done via the `EnvironmentManager`, which again will forward to registered receivers. One message can be sent to multiple receivers.

In the Smart Client environment, the messages are used extensively to notify about mode change, current camera selection as well as used to control many parts of the Smart Client.

The built-in messages have a name convention to assist in understanding which ones are used for issuing a command and which ones are received as indications. Also, where the data field is filled with a class or structure, it has a similar name as the command.

- Messages ending with "Command" – the sender want the receiver to perform something.
- Messages ending with "Indication" – the sender just want to inform about something, no action required.
- Messages ending with "Request" – the sender want the receiver to return a result.

For example the message with id "SmartClient.PlaybackCommand" can be sent by a plug-in to instruct the Smart Client to perform a specific playback command. The command to perform is defined by the `PlaybackCommandData` class and the message.Data is assigned this class.

```
EnvironmentManager.Instance.SendMessage(  
    new VideoOS.Platform.Messaging.Message(  
        MessageId.SmartClient.PlaybackCommand,  
        new PlaybackCommandData() { Command=PlaybackData.PlayForward,  
            Speed = _speed});
```

If `_speed` is set to a double e.g. 4.0 then this instruction will tell the Smart Client to start play in forward mode with speed 4 times normal. (This assumes that the Media is currently in Playback)

© Milestone Systems 2010



MIP SDK content

Samples



◆ Protocol

- ◆ Alarm Generator
- ◆ Status Console
- ◆ Transact Client
- ◆ Generic Event
- ◆ TCP Viewer

◆ Component

- ◆ Configuration Access
- ◆ Video Viewer
- ◆ Alarm Generator
- ◆ Central
- ◆ Image Viewer

◆ Plug-in

- ◆ Access Control
- ◆ Analytics Overlay
- ◆ Video Preview
- ◆ Video Replay
- ◆ Server Side Carousel
- ◆ Data Source
- ◆ Configuration Dump
- ◆ Service Test
- ◆ Message Tester
- ◆ Smart Client Window Tool

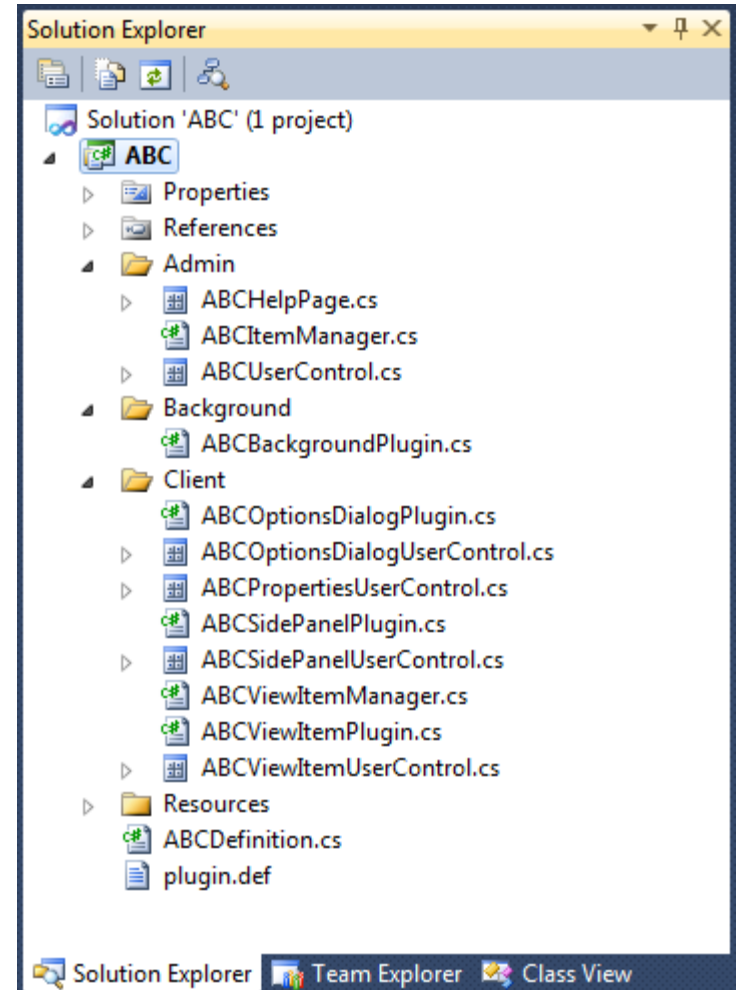


MIP SDK content

Visual Studio templates



- ◆ Available for
 - ◆ VS2008
 - ◆ VS2010
- ◆ Contains all key classes to get started
- ◆ Sample show how it looks when selecting name="ABC"
 - ◆ Class names modified
 - ◆ GUIDs generated
- ◆ Can build and execute directly

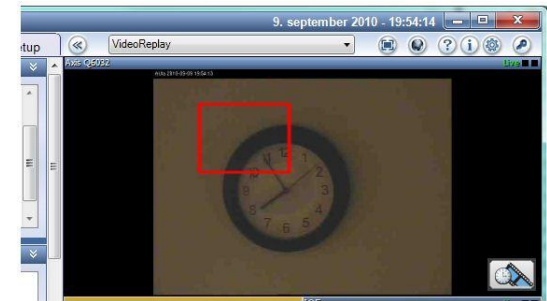


MIP SDK

Code sample - Video overlay



- ◆ Video overlay sample: Make a square on top of all available video being showed in the Smart Client
- ◆ Steps:
 - ◆ Create a new solution from template
 - ◆ Add code contained in appendix A (5 steps)
 - ◆ Total of apprx 75 lines of code
- ◆ Sample shows:
 - ◆ Abstraction from Complex ActiveX interface and versions
 - ◆ Same interface available in Smart Client and MIP .Net Library
 - ◆ Demonstrate how to 'hook' on to all video being showed in the Smart Client (Main window, floating window, print, bookmark, ...)





Summary

Jasleen Rehal



Milestone Integration Platform

A world of new possibilities



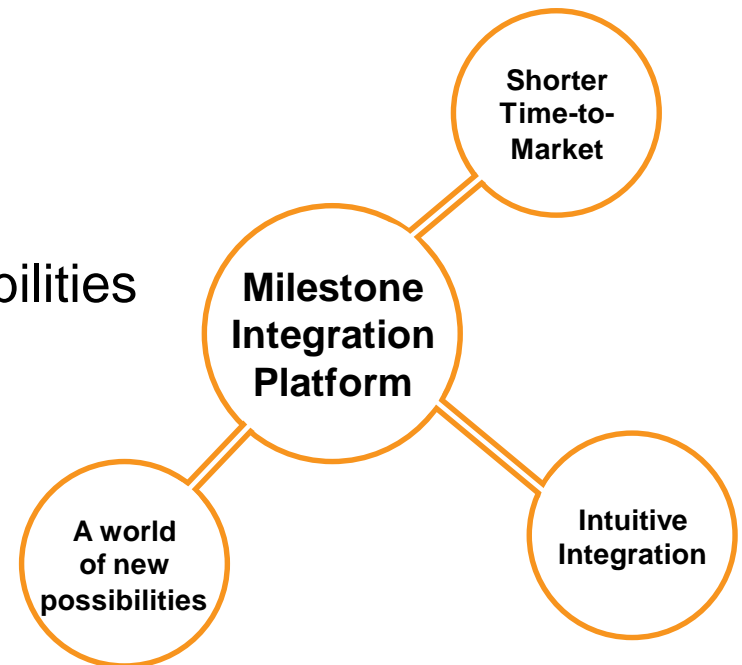
- ◆ Extensive development toolbox
 - ◆ Comprehensive documentation with application and code samples
 - ◆ Proven Milestone code libraries and components for common tasks: system logon, video decoding and rendering, configuration distribution, authorization verification, etc...
 - ◆ Development tool (traces, program logs, etc.)
 - ◆ Visual Studio project templates
- ◆ New and more powerful SDK capabilities
 - ◆ Integrated user management
 - ◆ Server side MIP integration
 - ◆ Richer event reporting
 - ◆ Configuration data repository
 - ◆ Dynamic video overlay



MIP – Key Benefits



- ◆ Intuitive Integration
 - ◆ MSP applications seamlessly integrated into plug-in architecture
- ◆ Shorter Time-to-Market
 - ◆ XProtect platform and release agnostic
 - *develop once - apply on all*
 - ◆ Extensive development toolbox
- ◆ A world of new possibilities
 - ◆ New and more powerful SDK capabilities



Getting started



- ◆ MIP SDK Beta is available for download from December 1st, 2010
 - ◆ Relevant information and link will be sent out via mail
- ◆ Sign up for new training sessions
 - ◆ In Copenhagen on January 24-25, 2011
 - ◆ In San Antonio, TX on February 28-March 1, 2011 - prior to Milestone Integration Platform Symposium, US



Key dates



- ◆ The MIP SDK 1.0 Beta is released on December 1st, 2010
 - ◆ Any integrations completed with the MIP SDK 1.0 Beta cannot be applied in commercial installations.
- ◆ The MIP SDK 1.0 is released in February 2011
- ◆ XProtect Enterprise 7.5 including MIP support
 - ◆ Beta – mid March 2011
 - ◆ Release - Q2 2011



Who to contact...



- ◆ Questions on MIP
 - ◆ Anders B. Christensen (abc@milestonesys.com)
- ◆ MIP SDK Training
 - ◆ Tine Elm (tel@milestonesys.com)
- ◆ MSP program
 - ◆ MSP team (partner@milestonesys.com)





Q&A

Milestone Systems - Confidential



The Open Platform Company

Appendix A

Code for overlay tutorial



- ◆ Create a new solution from template
- ◆ Add code for declaration & initialization
- ◆ Add code for managing registration
- ◆ Add code for add/remove new ImageViewerAddOn's
- ◆ Add code for camera id changing
- ◆ Add code for playback overlay update
- ◆ Add code for live overlay update
- ◆ Add code for bitmap construction



MIP Plug-in for analytics video overlay

Code sample - step 1



◆ Definition & Initialization

```
private Collection<ImageViewerAddOn> _activeImageViewerAddOns = new  
    Collection<ImageViewerAddOn>();
```

```
public override void Init()  
{  
    ClientControl.Instance.NewImageViewerControlEvent += NewImageViewerControlEvent;  
    _stop = false;  
    _thread = new Thread(new ThreadStart(Run));  
    _thread.Start();  
}
```

```
public override void Close()  
{  
    ClientControl.Instance.NewImageViewerControlEvent -= NewImageViewerControlEvent;  
    _stop = true;  
}
```



MIP Plug-in

Code sample - 2



◆ Registration

```
public override List<EnvironmentType> TargetEnvironments
{
    get { return new List<EnvironmentType>() { EnvironmentType.SmartClient }; }
}
```

```
void RegisterEvents(ImageViewerAddOn imageViewAddOn)
{
    imageViewAddOn.CloseEvent += ImageViewerAddOn_CloseEvent;
    imageViewAddOn.PropertyChangedEvent += ImageViewerAddOn_PropertyChangedEvent;
    imageViewAddOn.RecordedImageReceivedEvent += ImageViewerAddOn_RecordedImageReceivedEvent;
}

void UnregisterEvents(ImageViewerAddOn imageViewAddOn)
{
    imageViewAddOn.CloseEvent -= ImageViewerAddOn_CloseEvent;
    imageViewAddOn.PropertyChangedEvent -= ImageViewerAddOn_PropertyChangedEvent;
    imageViewAddOn.RecordedImageReceivedEvent -= ImageViewerAddOn_RecordedImageReceivedEvent;
}
```



MIP Plug-in

Code sample - 3



◆ Code for managing add/remove of new AddOns

```
void NewImageViewerControlEvent(ImageViewerAddOn imageViewAddOn)
{
    lock (_activeImageViewerAddOns)
    {
        RegisterEvents(imageViewerAddOn);
        _activeImageViewerAddOns.Add(imageViewerAddOn);
    }
}
void ImageViewerAddOn_CloseEvent(object sender, EventArgs e)
{
    ImageViewerAddOn imageViewAddOn = sender as ImageViewerAddOn;
    if (imageViewAddOn != null)
    {
        UnregisterEvents(imageViewerAddOn);
        if (_activeImageViewerAddOns.Contains(imageViewerAddOn))
        {
            lock (_activeImageViewerAddOns)
            {
                imageViewAddOn.ClearOverlay(12);
                _activeImageViewerAddOns.Remove(imageViewerAddOn);
            }
        }
    }
}
```



MIP Plug-in

Code sample - 4



◆ Maintenance & Playback overlay update

```
void ImageViewerAddOn_PropertyChangedEvent(object sender, EventArgs e)
{
    ImageViewerAddOn imageViewAddOn = sender as ImageViewerAddOn;
    if (imageViewAddOn != null)
    {
        imageViewAddOn.ClearOverlay(12);
    }
}

//Draw overlay in playback mode
void ImageViewerAddOn_RecordedImageReceivedEvent(object sender, RecordedImageReceivedEventArgs e)
{
    ImageViewerAddOn imageViewAddOn = sender as ImageViewerAddOn;
    if (imageViewAddOn != null)
    {
        DrawOverlay(imageViewerAddOn, imageViewAddOn.CameraFQID, e.DateTime);
    }
}
```



MIP Plug-in

Code sample -4



◆ Live update of overlay bitmap

```
private void Run()
{
    EnvironmentManager.Instance.Log(false, "ABC background thread", "Now starting...", null);
    while (!_stop)
    {
        if (_activeImageViewerAddOns.Count > 0)
        {
            lock (_activeImageViewerAddOns)
            {
                foreach (ImageViewerAddOn addOn in _activeImageViewerAddOns)
                {
                    if (addOn.CameraFQID != null && addOn.InLiveMode)
                    {
                        //Only draw the ones in Live mode
                        DrawOverlay(addOn, addOn.CameraFQID, DateTime.Now);
                    }
                }
            }
        }
        Thread.Sleep(1000);
    }
    EnvironmentManager.Instance.Log(false, "ABC background thread", "Now stopping...", null);
    _thread = null;
}
}
```



MIP Plug-in

Code sample -5



```
private void DrawOverlay(ImageViewerAddOn addOn, FQID cameraFQID, DateTime dateTime)
{
    Bitmap bitmap = ConstructBitmapOverlay(cameraFQID, dateTime);
    addOn.SetOverlay(bitmap, 12, false, true, true, 1.0, System.Windows.Forms.DockStyle.None,
        System.Windows.Forms.DockStyle.None, 0.0, 0.0);
    bitmap.Dispose();
}

private Bitmap ConstructBitmapOverlay(FQID cameraFQID, DateTime dateTime)
{
    Brush brush = new SolidBrush(Color.Transparent);
    Bitmap bitmap = new Bitmap(100, 100);
    Graphics g = Graphics.FromImage(bitmap);

    g.FillRegion(brush, new Region(new Rectangle(0, 0, 100, 100)));

    //Here we should add the real code to find relevant info for
    // CameraId=cameraFQID at time=dateTime
    int s = dateTime.Second % 10;
    g.DrawRectangle(Pens.Red, s * 5, s * 5, 30, 30);

    g.Dispose();
    return bitmap;
}
```

