

# WebALP 3.0

Iwane Laboratories, LTD.

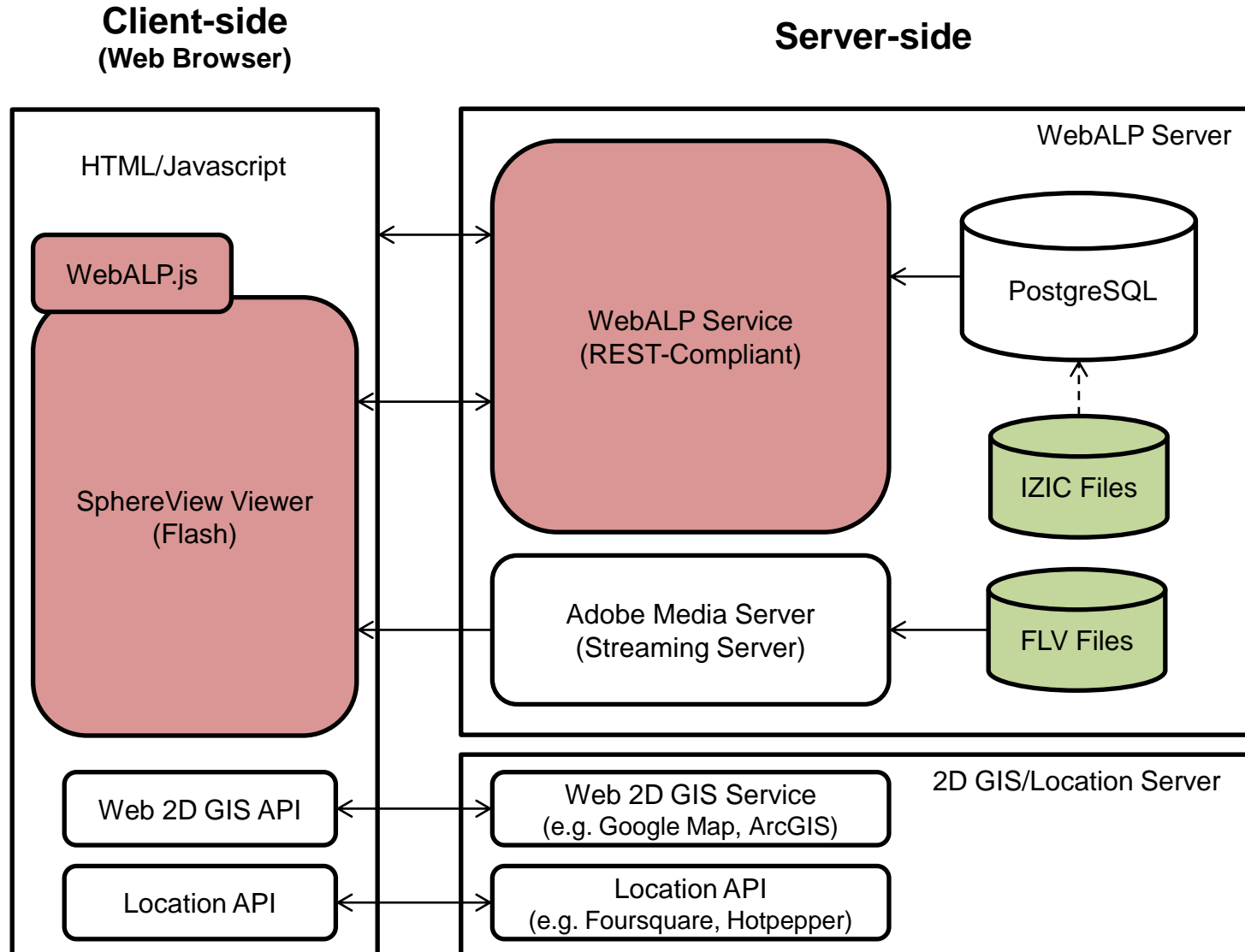
July 2014

# **INTRODUCTION TO WEBALP 3.0**

# What is WebALP?

- Frameworks to access 3D GIS data through the Internet.
  - Surrounding videos
  - Camera Vectors (Trajectory of camera position + orientation)
  - Knowledge of a 3D environment
  - Tagging and searching
- 3D Web GIS + Video streaming
  - Easy interface to mash with other web-service, e.g. 2D map.

# Where is WebALP in the Equation?



# Components of WebALP

- Main components (APIs)
  - WebALP Service (REST-Compliant)
  - SphereView (Flash) Viewer
    - Controllable from JavaScript API (though webALP.js)
- Utility components
  - IZICConverter
  - Data Maintenance Tools (DMT)

# Demo 1 : Web APP using WebALP

WebALP 3.0 for Chulalongkorn University - rev.22 -

Plugin Version : 3.0.1.35


MovieSegmentID : 15 FrameNo : 2130 Latitude : 13.66308844403981 Logitude : 100.792223110107

Stream ▾ Bitrate ▾ Play speed - x 1.0 ▾ Measurement - Auto ▾ Register CG Settings Toggle trajectory visibility Manual Switch layout

### Locations

Samut Prakan

[Rural Road](#)




**Latitude**  
13.6631693054576

**Longitude**  
100.792451652764

**Altitude**  
16.3673790621166

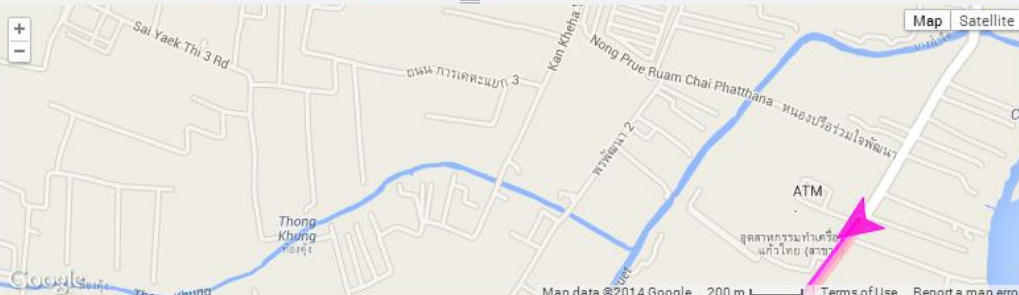
**Shooting date**  
2014-05-27 07:59:11



### Tag list

< prev 1 - 9 / 9 next >

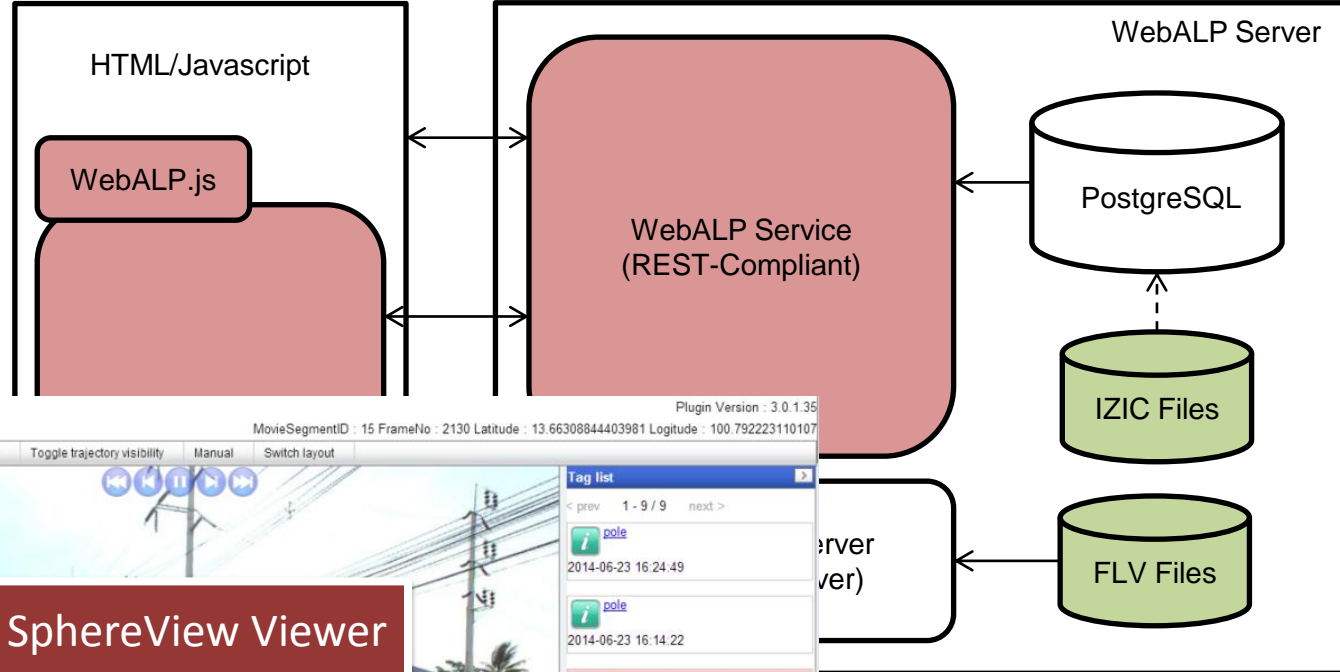
- [pole](#)  
2014-06-23 16:24:49
- [pole](#)  
2014-06-23 16:14:22
- [box](#)  
2014-06-23 16:13:40
- [pole3](#)  
2014-06-23 16:10:33
- [km benchmark](#)  
2014-06-23 16:06:20
- [รถยนต์ AUTO CAR](#)  
2014-06-23 15:18:46
- [ร้านก๋วยเตี๋ยว](#)  
2014-06-23 14:51:43
- [Thai Polymer Textile Co. LTD.](#)  
2014-06-23 14:50:13
- [MODEL\\_CONE1](#)  
2014-06-23 14:27:21



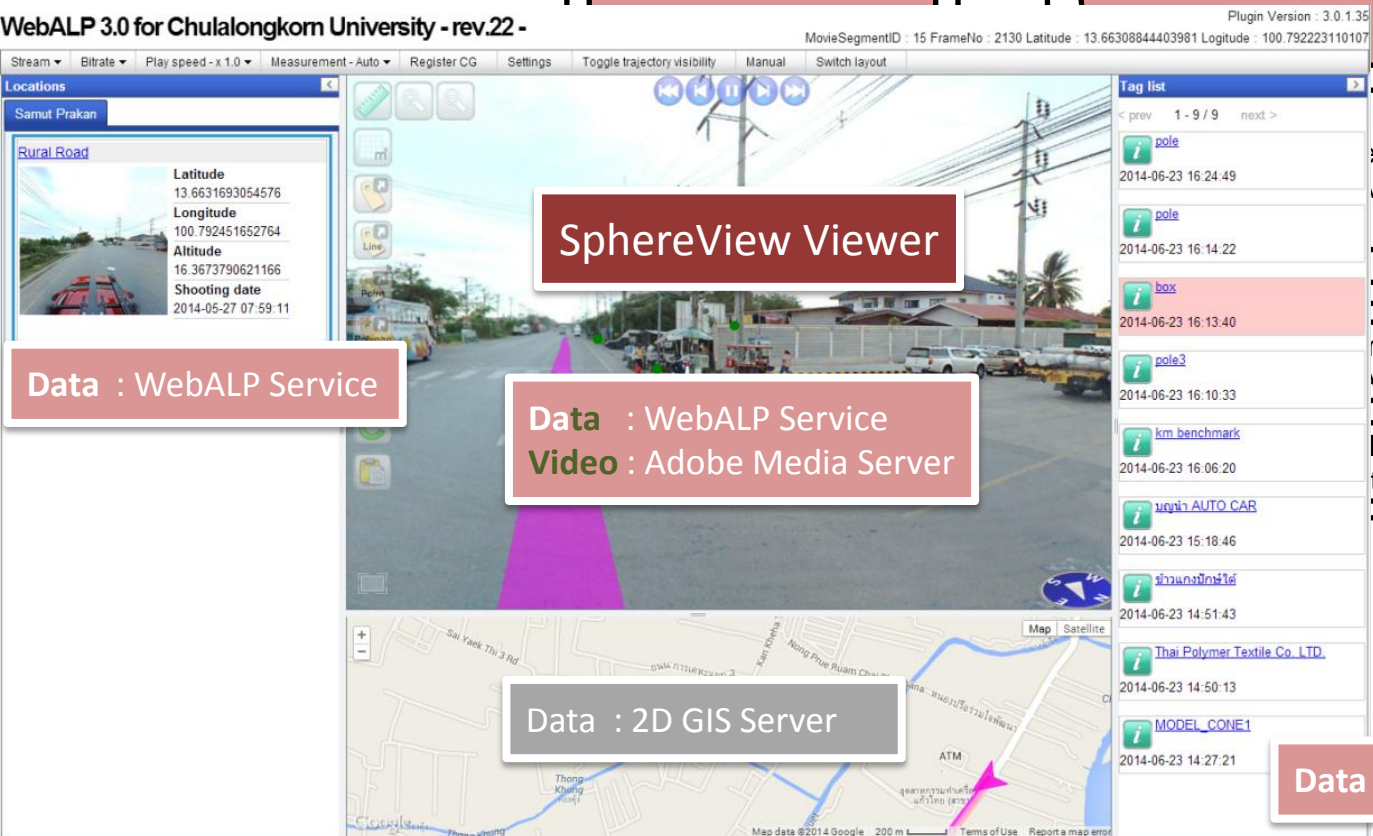
Map data ©2014 Google 200 m Terms of Use Report a map error

# Client-side (Web Browser)

# Server-side



WebALP 3.0 for Chulalongkorn University - rev.22 -



Data : WebALP Service

Data : WebALP Service  
Video : Adobe Media Server

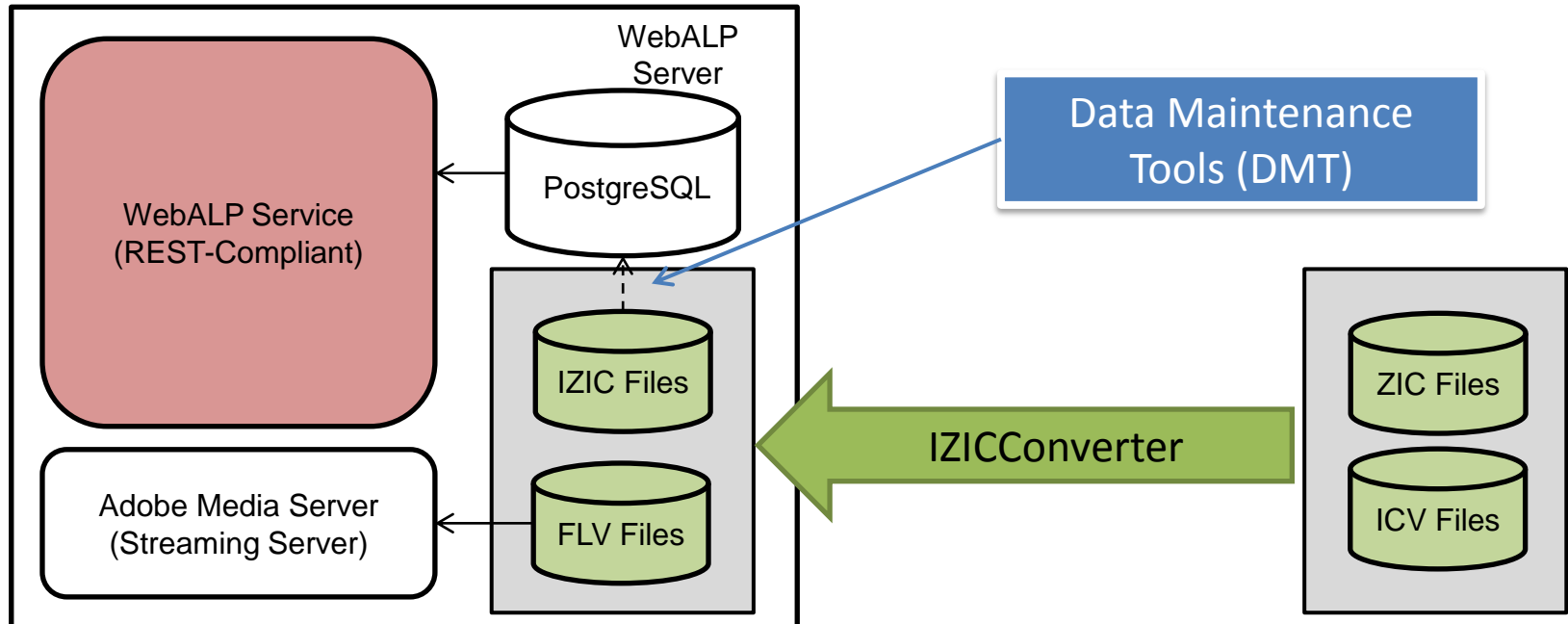
Data : 2D GIS Server

Data : WebALP Service

2D GIS/Location Server  
(ArcGIS)  
(OpenStreetMap)

# Utility Components

## Server-side





# Demo : IZICConverter

The screenshot shows the IZICConverter application window. The title bar reads "IZICConverter". The menu bar includes "File(F)" and "Help(H)".

The "Task List" section contains a table with the following data:

	srcZIC	srcICV	imageWidth	imageHeight	frameNum	cameraNum	divideNum	frameRate
▶	C:\Users\pvmilk...	C:\Users\pvmilk...	5400	2700	993	2	1, 3	16.00256
	C:\Users\pvmilk...	C:\Users\pvmilk...	5400	2700	895	2	1, 3	16.00256

Below the table is a "Settings" section with the following fields:

- Output Directory(O): C:\Users\pvmilk\Desktop\output
- JPEG Quality(J): 95 %
- Thread Number(T): 4

The "Command line After Convert" section shows the following command line:

```
"C:\cubeconv\webalp3\create_cube.bat" "$(dstCubeFrontForwardStreamDir)" "$(outputDirectory)\stream" "$frameRate" "$fileName_cube_front_forward" "C:\cubeconv\webalp3\create_cube.bat" "$(dstCubeRearReverseStreamDir)" "$(outputDirectory)\stream" "$frameRate" "$fileName_cube_rear_reverse" "C:\cubeconv\webalp3\create_cube_all.bat" "$(dstCubeAllForwardStreamDir)" "$(outputDirectory)\stream" "$frameRate" "$fileName_cube_all_forward" "C:\cubeconv\webalp3\create_cube_all.bat" "$(dstCubeAllReverseStreamDir)" "$(outputDirectory)\stream" "$frameRate" "$fileName_cube_all_reverse"
```

The "Parameters(P):" section lists the following parameters:

```
$(srcZIC) $(srcICV) $(imageWidth) $(imageHeight) $(cameraNum) $(frameNum) $(frameRate) $(dstZIC) $(outputDirectory) $(fileName) $(dstCubeFrontForwardStreamDir) $(dstCubeRearReverseStreamDir) $(dstCubeAllForwardStreamDir) $(dstCubeAllReverseStreamDir)
```

At the bottom right, there are "Convert(E)" and "Abort(B)" buttons. The status bar at the bottom left shows "Speed: 1.796 frames/s , Total progress: 0/2".

# Demo : Data Maintenance Tools (DMT)

**Data Maintenance Tool 3.0** Service Version : 3.0.2.3  
Project ID : 1 Route ID : 22 Class ID : 20 Group ID : 32 Movie segment ID : 628 Movie segment name : sapporo\_2011-0718-103327\_A\_0002.izic Frame No : 58 M : 0  
Latitude : 43.08864236852531 Longitude : 141.331431105558 Altitude : 13.420214894221  
Stream URL : ---

Project SphereView Resource Configure Tag Stream - Front/Rear view Btrate - 1024 kbps Play speed - x1 Measurement - Auto

OpenStreetMap contributors

**Project**

**Project tree**

- HongKong
- 岩根研究所
  - IzicConveter1.0.0.63
    - Main line
    - Down
    - test
- 子々七オンテータ
- 札幌駅前
- 名古屋高速道路協会

**Disconnected movie segments**  
The total number of items:53

sapporo_2011-0718-1111116_B_0004.izic(614)	show / edit / delete
sapporo_2011-0718-1111116_B_0003.izic(615)	show / edit / delete
sapporo_2011-0718-1111116_B_0002.izic(616)	show / edit / delete
sapporo_2011-0718-1111116_B_0001.izic(617)	show / edit / delete
sapporo_2012-1121-133158_J_0001.izic(578)	show / edit / delete
Sapporo_2012-1121-133158_M_0002_cut2.izic(573)	show / edit / delete
2014-0508-143323_comp_0001.izic(584)	show / edit / delete
Sapporo_2012-1121-133158_J_0001.izic(578)	show / edit / delete

**Connected movie segments**  
The total number of items:11

sapporo_2011-0718-103327_A_0002.izic(628)	show / edit / delete
sapporo_2011-0718-104305_A_0001.izic(627)	show / edit / delete
sapporo_2011-0718-104305_A_0002.izic(626)	show / edit / delete
sapporo_2011-0718-104305_A_0003.izic(625)	show / edit / delete
sapporo_2011-0718-104305_A_0004.izic(624)	show / edit / delete
sapporo_2011-0718-104305_A_0005.izic(623)	show / edit / delete
sapporo_2011-0718-104305_A_0006.izic(622)	show / edit / delete
sapporo_2011-0718-104305_A_0007.izic(621)	show / edit / delete

岩根研究所 [projectID = 1] = IzicConveter1.0.0.63 [routeID = 22] > class type Main line [classID = 20] > Down [groupID = 32]

Reload Close

SphereView ver 3.0.1.35

Copyright 2014 Iwano Laboratories, Ltd. All rights reserved. (SRevision: 897 S)

# Outline of Training Session

- WebALP 3.0 Installation/Setup
- Preparing data for WebALP 3.0
- WebALP Service API
- JavaScript API for SphereView
  - SphereView

# **WEBALP 3.0 INSTALLATION/SETUP**

# System Requirements

- Operating System
  - Windows Server 2008 R2; x64
  - Windows Server 2012; x64
  - Windows Server 2012 R2; x64
- Recommended Specification
  - CPU : Intel Xeon Processor E5-1410 2.8GHZ
  - Memory : 8 GB
  - Harddisk : < 1GB for WebALP3 Service

# Pre-requisite Softwares

- \*IIS 7.5 ASP.NET or IIS8.0 ASP.NET4.5
- Microsoft .NET 4.5 or higher
- Visual c++ redistributable for visual studio 2012
- PostgreSQL 9.3.X + PostGIS 2.1.X
- Sentinel HASP/LDK 6.63 Run-time (dongle license)
- \*Adobe Media Server 5

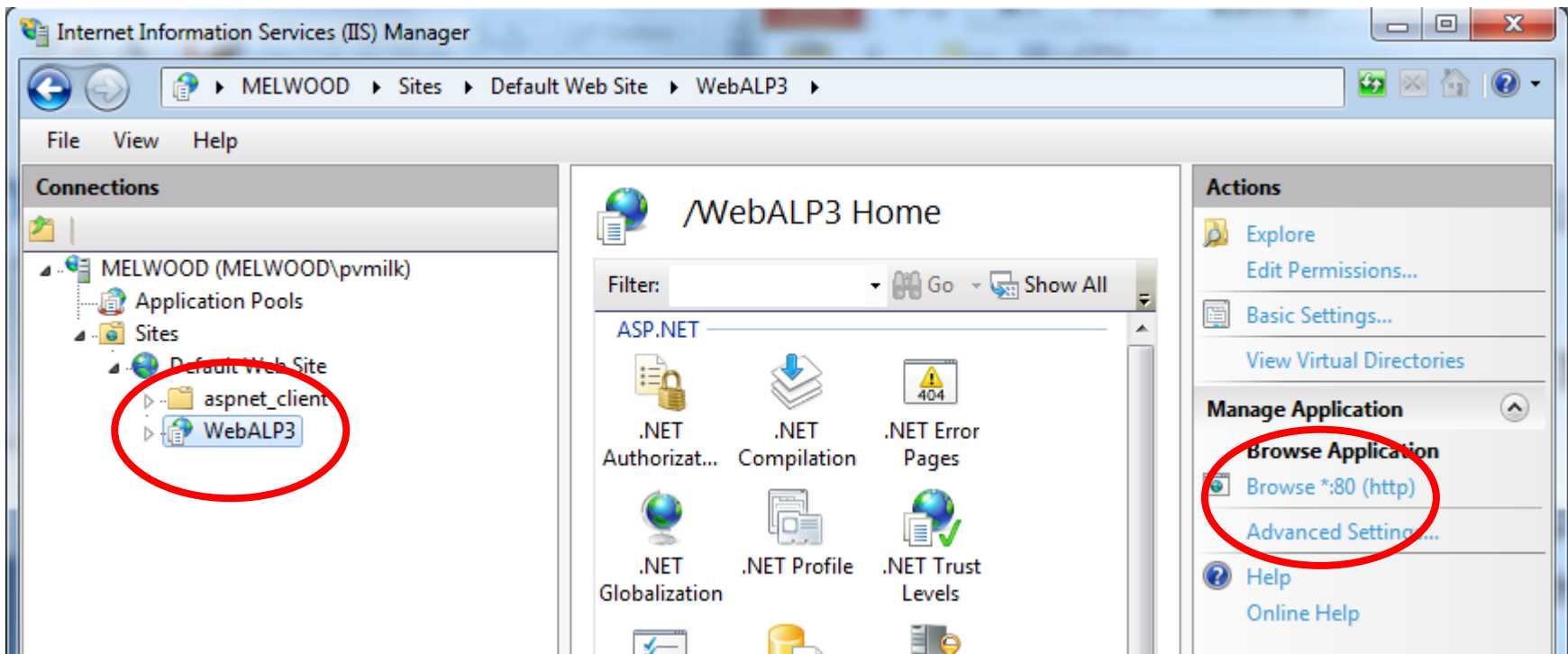
\*Not included in a installer-bundle.

# Installation

- Follow the on-screen instruction of the installer bundle.
  - “Adobe Media Server” and “Postgres/PostGIS” can be installed in a separated machine.
- Careful :
  - When installing PostGIS 2.1.X, make sure to install the option "**Create spatial database**".

# Confirming an Installation

- Browse to the page,
  - If successful, the WebALP3 welcome page should appear.





# WebALP3 Welcome Page

← → ↻ 🏠 localhost/WebALP3/

## Welcome to WebALP3.0!

### Contents List

- [DMT](#)
- [Sample Code Gallery](#)
- [Documents](#)

### Service Status

Error - C:\inetpub\wwwroot\WebALP3\App\_Data\license.dat not found.

### License

Input license.dat :  No file chosen

# Activating WebALP3

- A correct license (license.dat) file must be uploaded to WebALP3 before it can be used.
  - Using upload function in the Welcome page.

# Configure DMT

- DMT-> Configure

Configuration

\*File root directory: C:\

\*Context root url: http://192.168.10.156/webalp3/

\*IZIC connection

pool size: 20

\*Keep alive interval: 60000

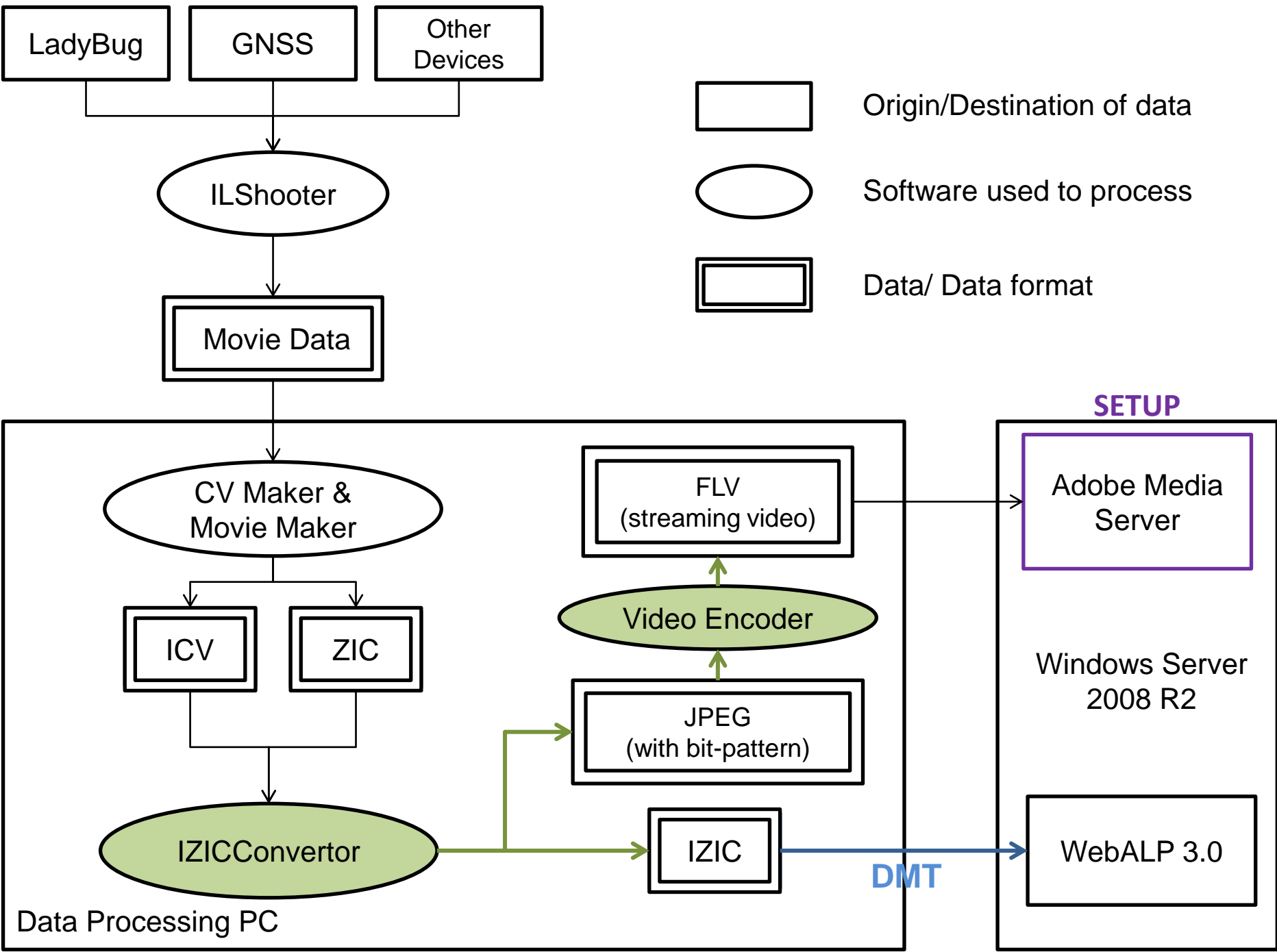
Result limit

\*Tag: 100

\*Movie segment: 100

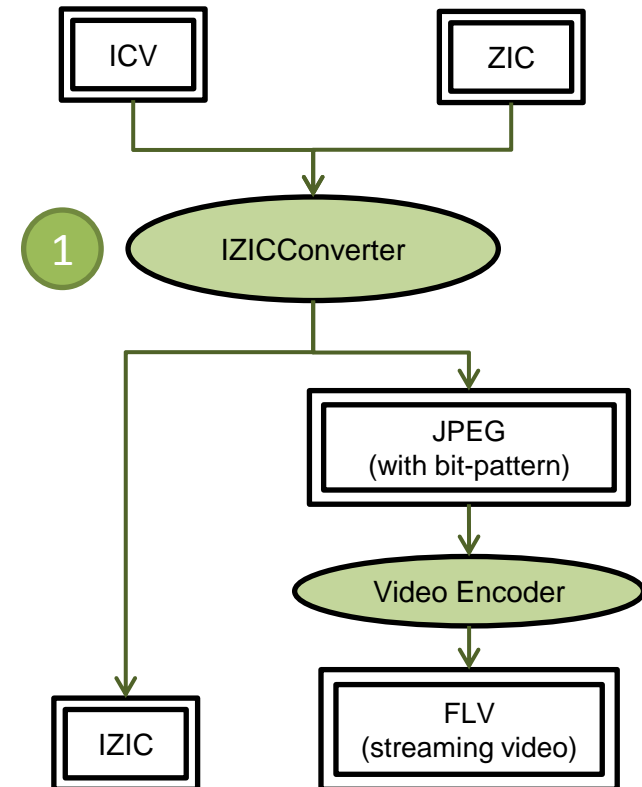
OK Cancel

# **PREPARING DATA FOR WEBALP 3.0**



# IZICConverter

- Divides a spherical image into multiple perspective images suitable for streaming.
- Input
  - ICV + ZIC
- Output
  - IZIC
  - JPEG (with bit pattern)



# Example of FLV files



# What does IZICConverter do?

[Image data]

Spherical Image (ZIC)



Perspective Image

Front



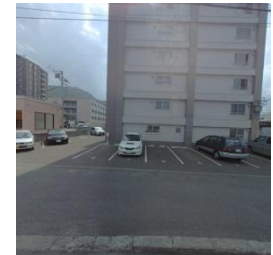
Rear



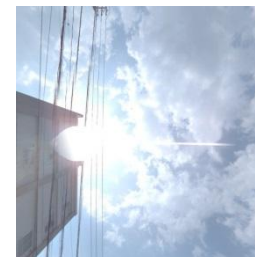
Right



Left



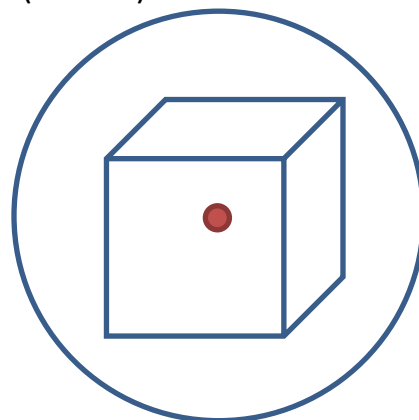
Top



Bottom



- 1.) Map a spherical image to a sphere
- 2.) Create perspective images (6 sides)

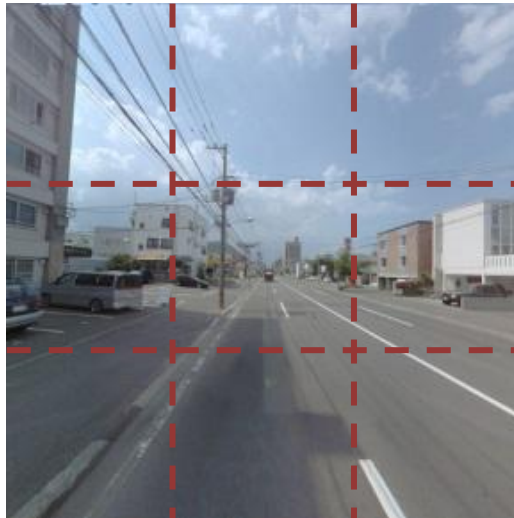




# IZIC File

- All perspectives images
  - further divide into smaller patches (512x512 px) suitable for transferring through the Internet, e.g.
- Thumbnails version of a spherical image.
  - For instant preview.

Front



# JPEG with Bit-Pattern (16 bits)

- These JPEG images are used to create streaming videos.
  - Bit pattern is used for seeking an exact video frame during playback.



Front view of Frame 398<sup>th</sup>  
(CUBE FRONT)



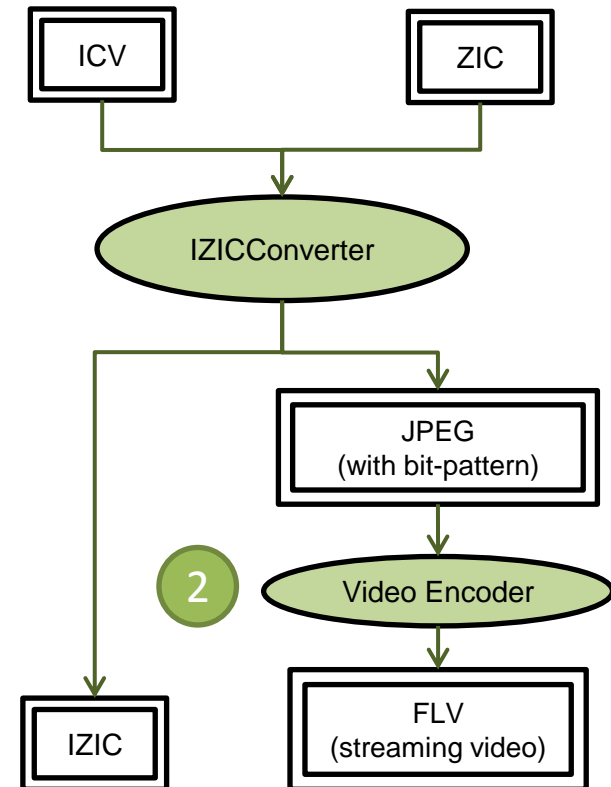
All view of Frame 114<sup>th</sup>  
(CUBE ALL)

# DEMO:

- Check folder structure
- Input : Sapporo0001.zic + Sapporo0001.icv
- Output :
  - Folder \$OUT\_DIR/Sapporo0001
    - cube
      - forward
      - reverse
    - cubeALL
      - forward
      - reverse

# Video Encoder

- Combine images into a streaming video.
- Support format : FLV
- Input
  - JPEG with bit pattern
- Output
  - FLV streaming video



# Currently Supported Videos

- Camera-directional view (cube)
  - (FRONT) Forward
  - (REAR) Reverse
  - \*Higher quality, with the same bit-rate
- 360 view (cubeALL)
  - Forward
  - Reverse
  - \*Lower quality, with the same bit-rate.
- DEMO

# Creating FLV using IZICConverter + FFMPEG

- pre-requisite : ffmpeg installed (callable in cmd.exe)
  - Other video encoder can be used as well.
- IZICConverter allows you to call command-line after each file conversion is finished.

Command line After Convert

Command line(A):

```
"C:\%cubeconv%\webalp3\create_cube.bat" "$(dstCubeFrontForwardStreamDir)" "$(outputDirectory)\%stream" "$(frameRate)" "$(fileName)_cube_front_forward"  
"C:\%cubeconv%\webalp3\create_cube.bat" "$(dstCubeRearReverseStreamDir)" "$(outputDirectory)\%stream" "$(frameRate)" "$(fileName)_cube_rear_reverse"  
"C:\%cubeconv%\webalp3\create_cube_all.bat" "$(dstCubeAllForwardStreamDir)" "$(outputDirectory)\%stream" "$(frameRate)" "$(fileName)_cube_all_forward"  
"C:\%cubeconv%\webalp3\create_cube_all.bat" "$(dstCubeAllReverseStreamDir)" "$(outputDirectory)\%stream" "$(frameRate)" "$(fileName)_cube_all_reverse"
```

Parameters(P):

```
$(srcZIC) $(srcICV) $(imageWidth) $(imageHeight) $(cameraNum) $(frameNum) $(frameRate) $(dstIZIC) $(outputDirectory) $(fileName)  
$(dstCubeFrontForwardStreamDir) $(dstCubeRearReverseStreamDir) $(dstCubeAllForwardStreamDir) $(dstCubeAllReverseStreamDir)
```

# A Naming Method for FLV

- Must be corresponded with SphereView (Javascript).
  - WebALP.MovieController.SetStreamConfig(StreamConfig config)
    - streamConfig:forwardProjectType = `FORWARDTYPE` = CUBE\_FRONT
    - streamConfig:reverseProjectType = `REVERSETYPE` = CUBE\_ALL
    - streamConfig:suffix = `SUFFIX` = "2048k"
    - streamConfig:streamFileType = `FILETYPE` = FLV
  - For example, Sapporo0001.izic
    - Template Stream Name = `STREAMTEMPLATE` = Sapporo0001
    - (Default; This value can be modified in DMT)
- Forward play button
  - lowercase(`STREAMTEMPLATE`\_`FORWARDTYPE`\_forward\_`SUFFIX`\_.`FILETYPE`)
  - sapporo0001\_cube\_front\_forward\_2048k.flv
- Reverse play button
  - lowercase(`STREAMTEMPLATE`\_`REVERSETYPE`\_reverse\_`SUFFIX`\_.`FILETYPE`)
  - sapporo0001\_cube\_all\_reverse\_2048k.flv

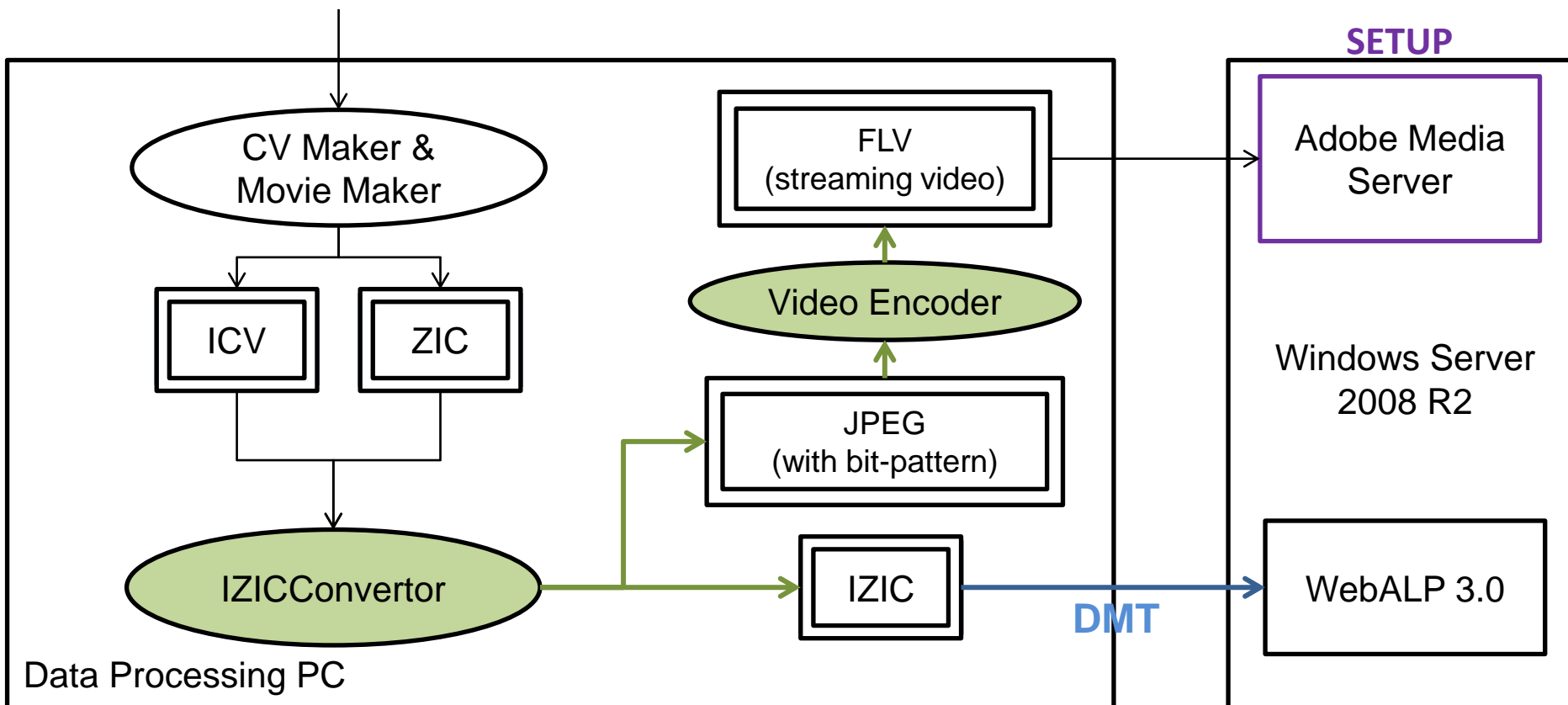
# DEMO:

- Creating IZIC and FLV in one go.
- Check out \*.bat file.



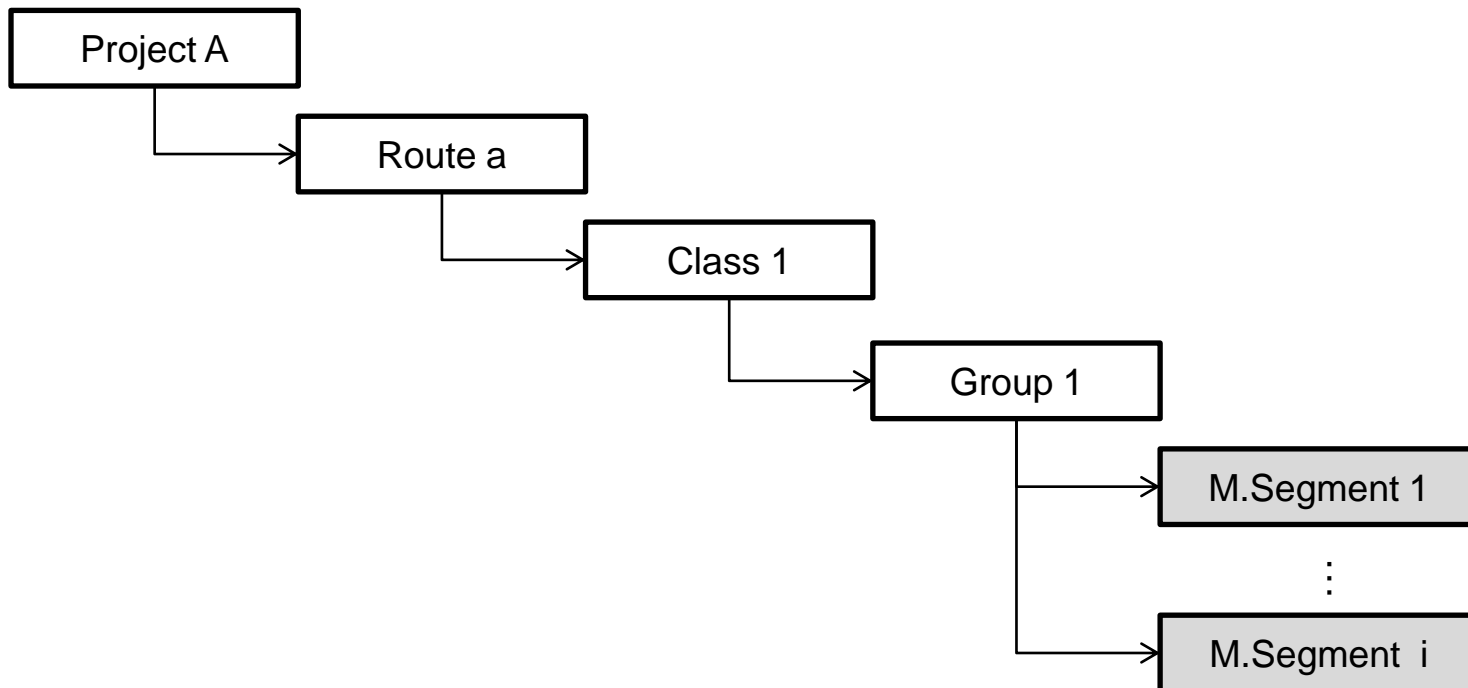
# Data Maintenance Tools (DMT)

- Registering data to the system
  - Image + Video data (Can only be done HERE).
  - Tag data (Can also be done by users through WebALP Service).



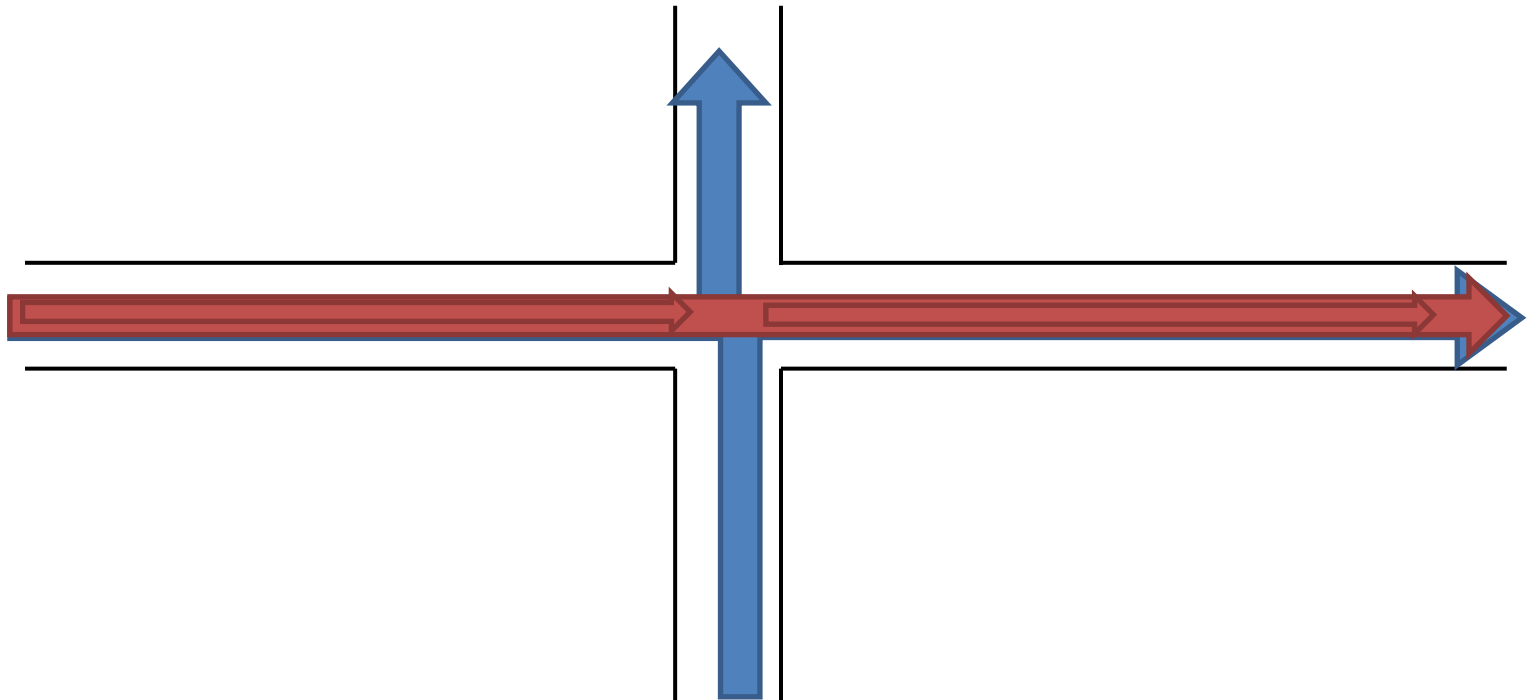
# Structure of Movie Data in WebALP3

- IZIC file is referred to as `Movie`.
- `Movie Segment` is created from `Movie` by specifying a begin/end frame.
  - Movie Segment from the same Movie can be added to multiple group



# Connecting Movie Segments

- More than one movie segments can be connected, in order to play a streaming video continuously
  - Similar to a playlist.



# DEMO: Register Movie in DMT

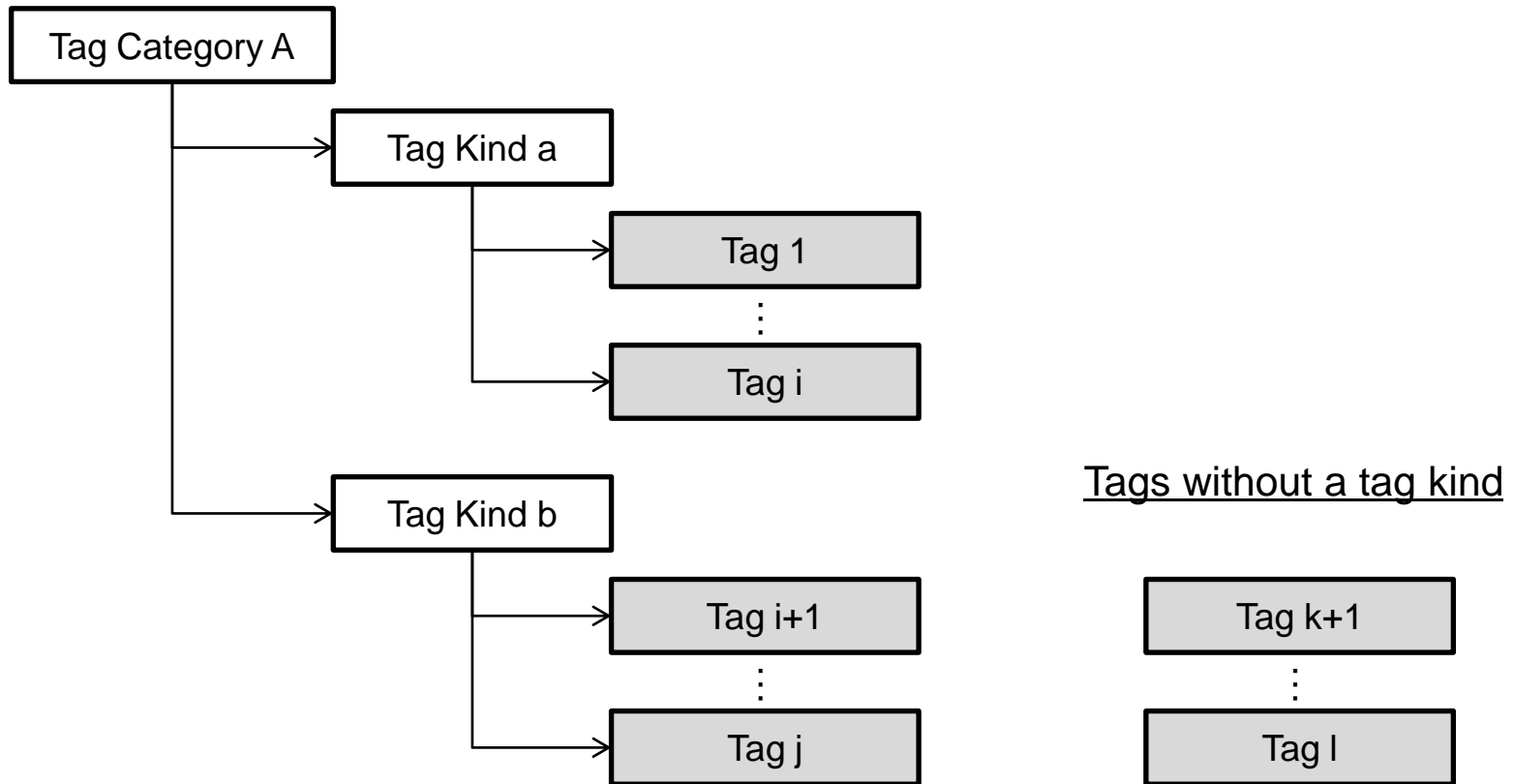
- Make sure the resource folder has a read/write permission for IIS Users (or group IIS\_IUSRS)
- Note to mention :
  - Class : Only 3 types
  - Group : Only 2 types

The screenshot shows a dialog box titled "Add resource path" with a close button (X) in the top right corner. The dialog is divided into several sections:

- \*Name:** A text box containing "Champion Data".
- Image directory:** A section containing two text boxes:
  - \*Source:** "C:\webalp3\_data\izic"
  - \*Destination:** "C:\webalp3\_data\izic\output"
- Stream:** A section containing two text boxes:
  - \*Server URL:** "rtmp://192.168.10.156"
  - \*VOD root:** "vod\_webalp3"

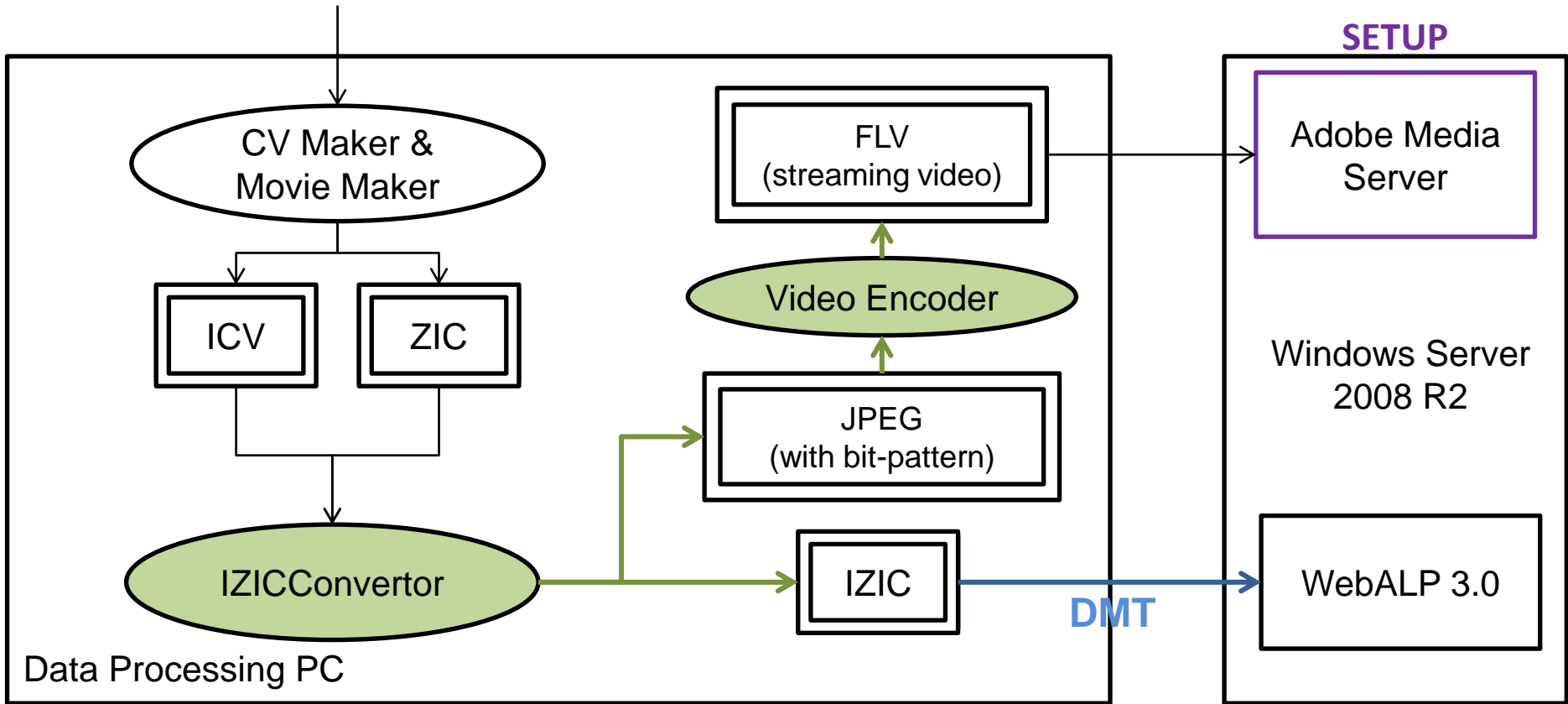
At the bottom right of the dialog, there are two buttons: "Submit" (highlighted in blue) and "Cancel".

# Structure of Tag Data in WebALP3



# DEMO: Register Tag in DMT

# Setup Adobe Media Server



# Adobe Media Server Starter

- Install option
  - No need to install Apache 2.2
  - Make sure there is no port crashing



# Add New VOD Application (1)

- Create new on-demand video application
  - Copy \$AMS\_HOME/samples/applications/vod
  - Rename the folder to the desired application name e.g. **vod\_webalp3**
  - Move the folder to \$AMS\_HOME/applications
- Specify folder of the media files :
  - Edit \$AMS\_HOME/applications/**vod\_webalp3**/Application.xml
    - Replace <Streams>/;\${VOD\_COMMON\_DIR}</Streams> <Streams>...<Streams>
    - With <Streams>/;C:\webalp3\_data\stream<Streams>
    - \* Multiple folder can be specified here.

# Add New VOD Application (2)

- Checking result
  - Administrator Console
    - \$AMS\_HOME/webroot/index.html
    - Login -> Manage Servers -> Applications
  - Streaming FILE in RTMP Player (locally)
    - <http://www.ideaweb.it/eng/player.cfm>
    - Access URL : rtmp://localhost/vod\_webalp3/FILENAME (without .flv)
- Note
  - To make this accessible from other machines; please edit the firewall configuration.
    - TCP/UDP of Inbound and Outbound rtmp port

# Additional Configuration for WebALP3

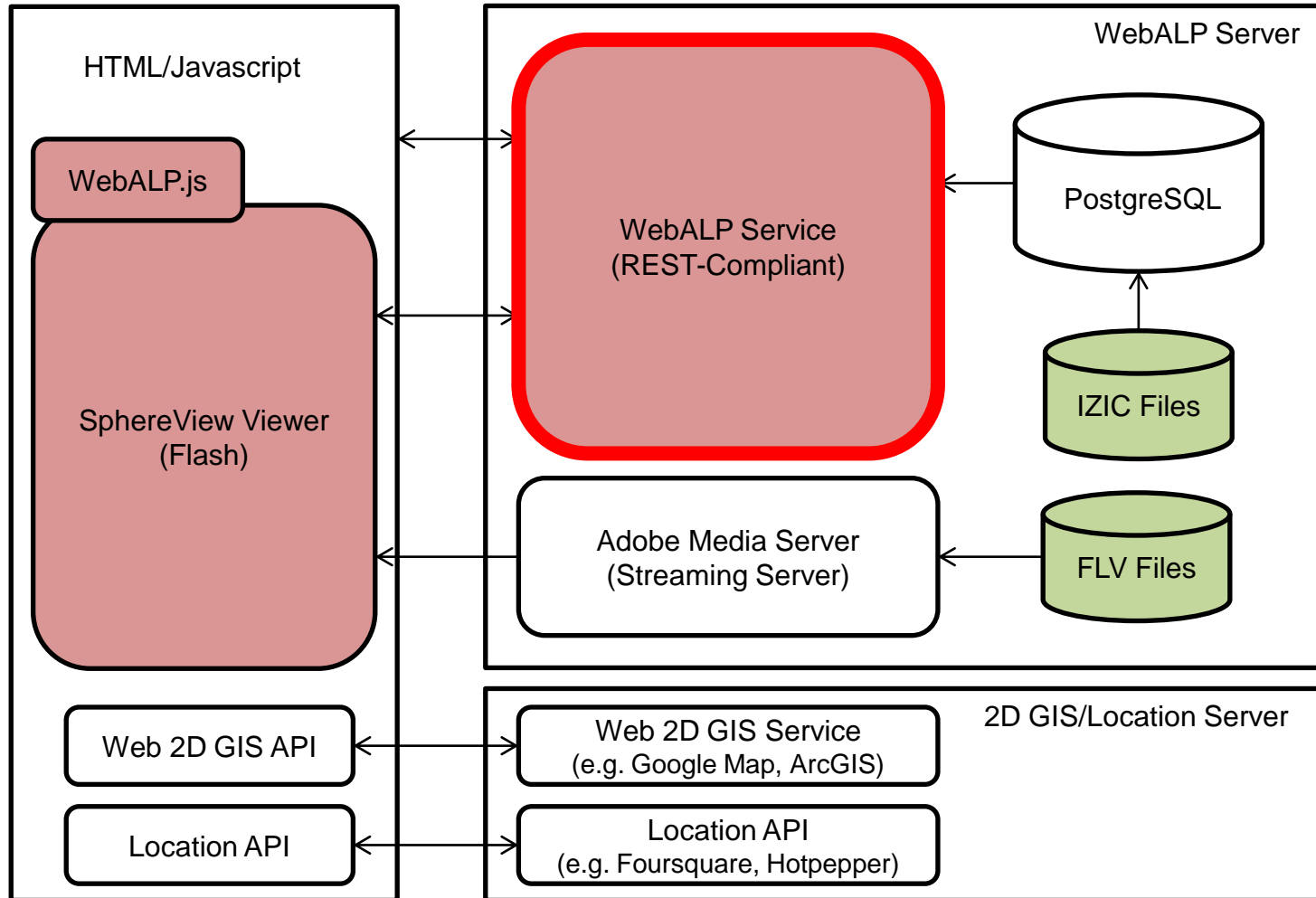
- `$AMS_HOME/conf/_defaultRoot/_defaultVHost/Application.xml`
  - Allow streaming video to be modified on the fly.
    - `<FolderAccess>true</FolderAccess>`
    - `<AudioSampleAccess enabled="true"/></AudioSampleAccess>`
    - `<VideoSampleAccess enabled="true"/></VideoSampleAccess>`
  - Meta-Data to sent once during Seek.
    - `<SendDuplicateOnMetaData>>false</SendDuplicateOnMetaData>` (Default is true)
- Note :
  - Don't forget to restart the media server before proceed.

# DEMO: Streaming Video in DMT

# **WEBALP SERVICE API**

**Client-side  
(Web Browser)**

**Server-side**



# WebALP Service API

- Web Service Handler
  - Access URL : \$CONTEXT\_ROOT\_URL/service.asmx
    - <http://192.168.10.156/webalp3/service.asmx>
  - GET/POST
- Page Handler
  - AddTagKind | UpdateTagKind | RegisterTag
  - Access URL : \$CONTEXT\_ROOT\_URL/AddTagKind.ashx
  - Access URL : \$CONTEXT\_ROOT\_URL/UpdateTagKind.ashx
  - Access URL : \$CONTEXT\_ROOT\_URL/RegisterTag.ashx
  - GET :If binary data is not required.
  - POST :when the uploading of binary data is required.
- Return : XML Format

# Token ID and Password

- Authentication and keep track of # of users
- Obtainable from SphereView API
  - `WebALP.SphereView.GetTokenID()`
    - Constant in a particular session.
  - `WebALP.SphereView.GetPassword()`
    - Please refresh every time you call any method.



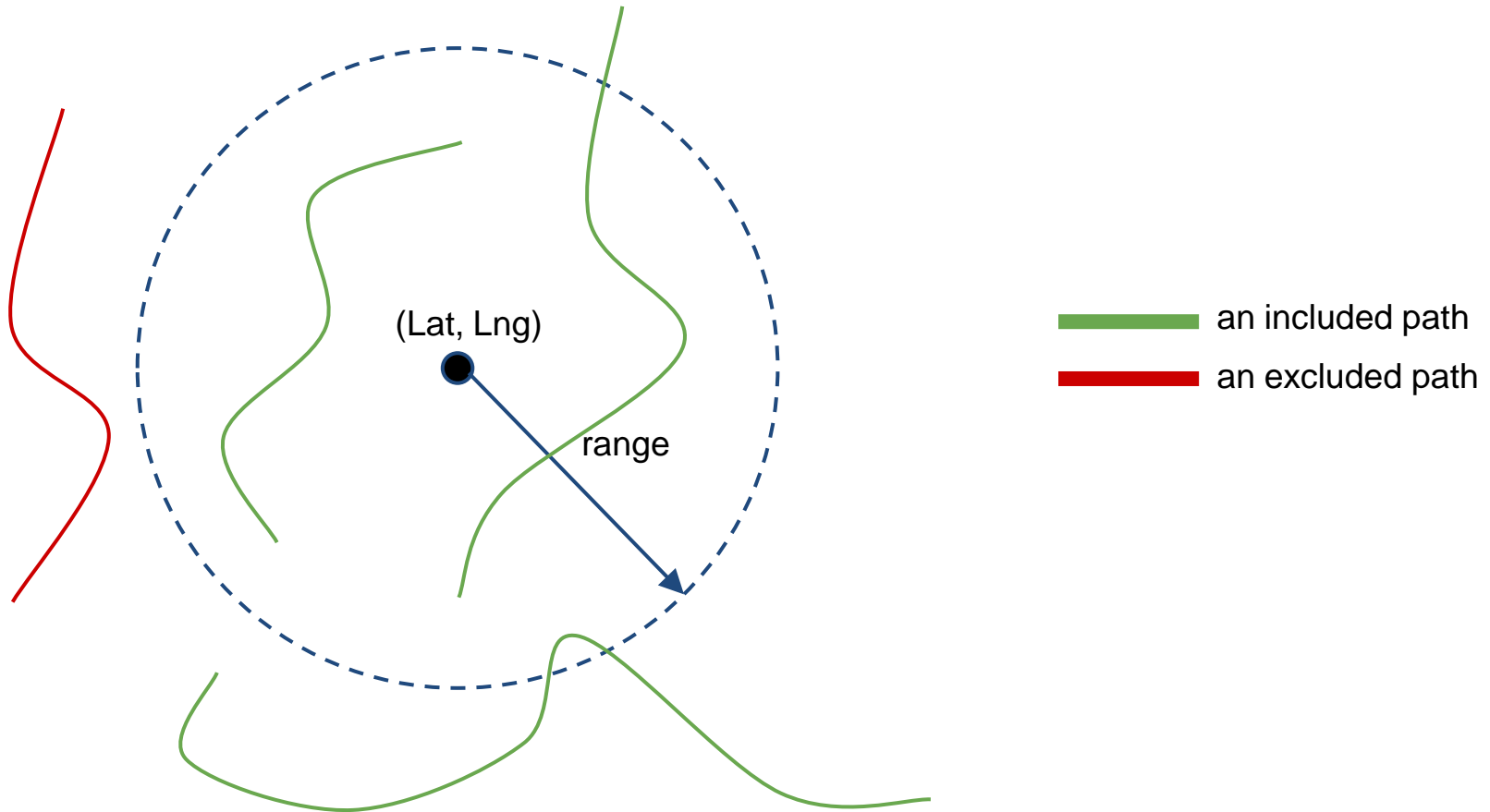
# Methods Need Clarification

- `WebALP3::Service::GetAroundMovieSegmentPath`
- `WebALP3::RegisterTag`
- `WebALP3::Service::GetTag`
- `WebALP3::Service::GetTagByPaging`

# GetAroundMovieSegmentPath

- Get nearby movie segment paths by specified conditions
  - E.g. For overlaying movie paths in a 2D map.
- Input :
  - ...
  - range
  - Interval

# GetAroundMovieSegmentPath [parameter : range]

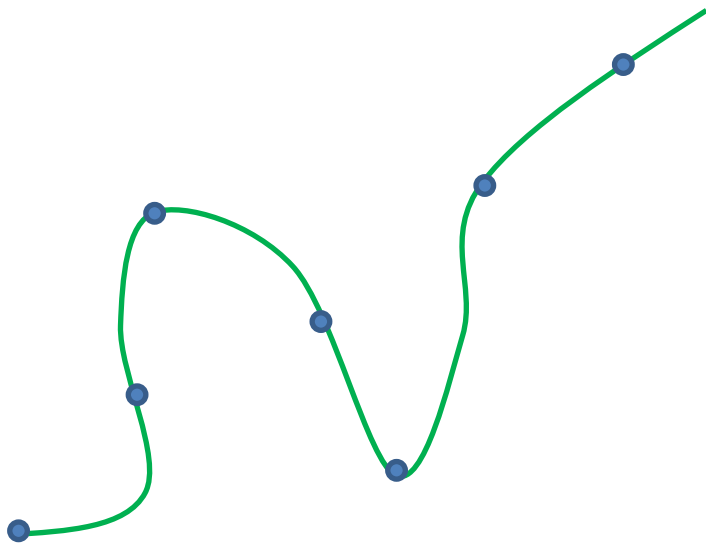


When the path is included, the whole trajectory will be returned (not only those within the range).

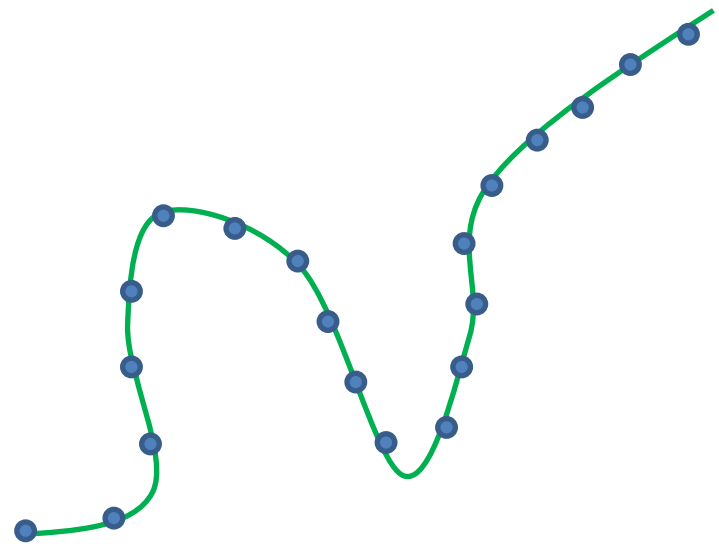
# GetAroundMovieSegmentPath

[parameter : interval]

- The more the value, the rougher the camera trajectory would be returned.
  - See more : tolerant value in a Douglas-Peucker algorithm.



Interval = 100



Interval = 60

# RegisterTag

- Use to add/update tags in the system.
- Input :
  - ...
  - tagXML
  - Signature
  - attachedFile (binary)

# Add/Update with RegisterTag

- Updating when `id` attribute is specified; otherwise adding.
- Multiple tags can be added/updated at the same time through one web-service call.

```
<tagSet xmlns="http://www.iwane.com/ALV/">
  <tag >
    ...
  </tag>
  ...
  <tag id="27680" >
    ...
  </tag>
</tagSet>
```

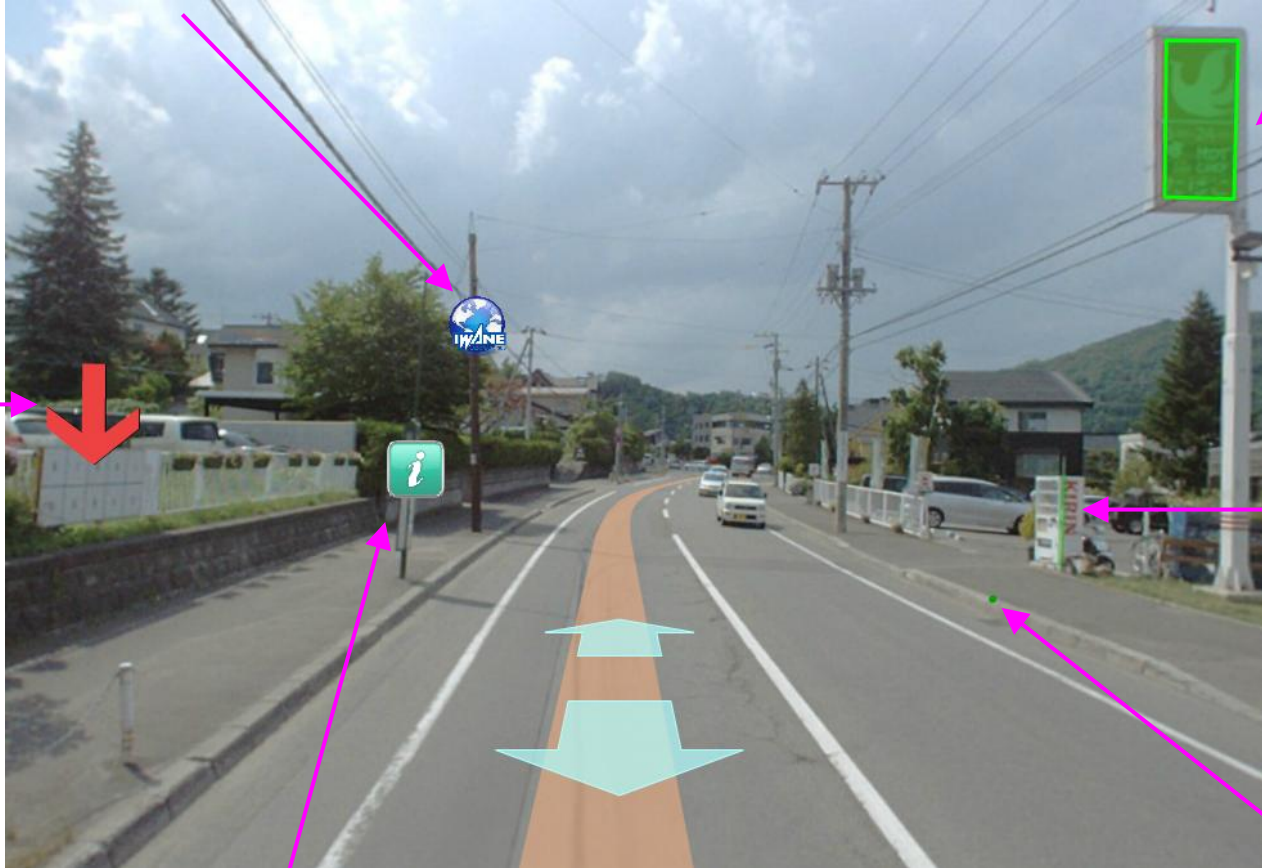
Adding Tag

Updating Tag

# Tag and its Appearance

ICON with icon image or tag kind's icon image

POLYGON



MODEL

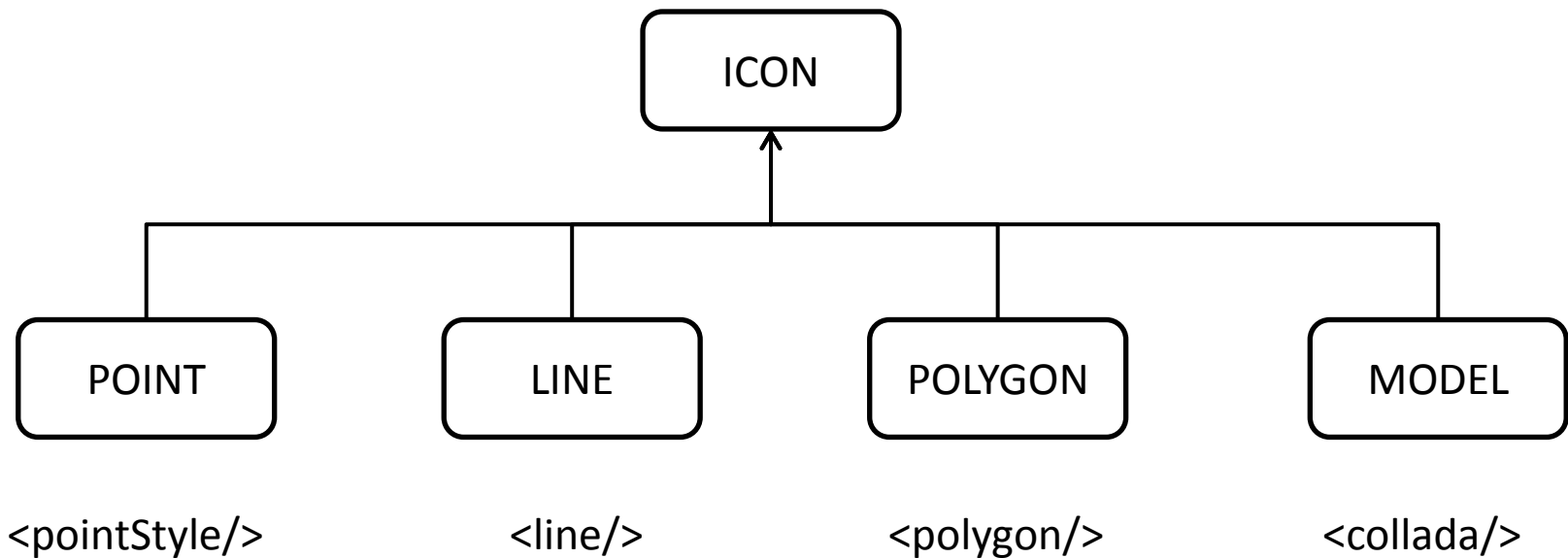
LINE

ICON with default image

POINT

# TagXML

- Namespace : "http://www.iwane.com/ALV/"
- Different based on a type of tag.



- Custom element within TagXML.



# Signature

- Additional field for developer to attach with a tag.
  - Can be used to verify the ownership.
  - If set during adding, the same must be provided when updating or deleting.
- Optional

# Tag's Attachment File

- Webservice.RegisterTag
  - Optional parameters
- File can be in any format;
  - File will upload to the WebALP server.
- Access URL is stored in

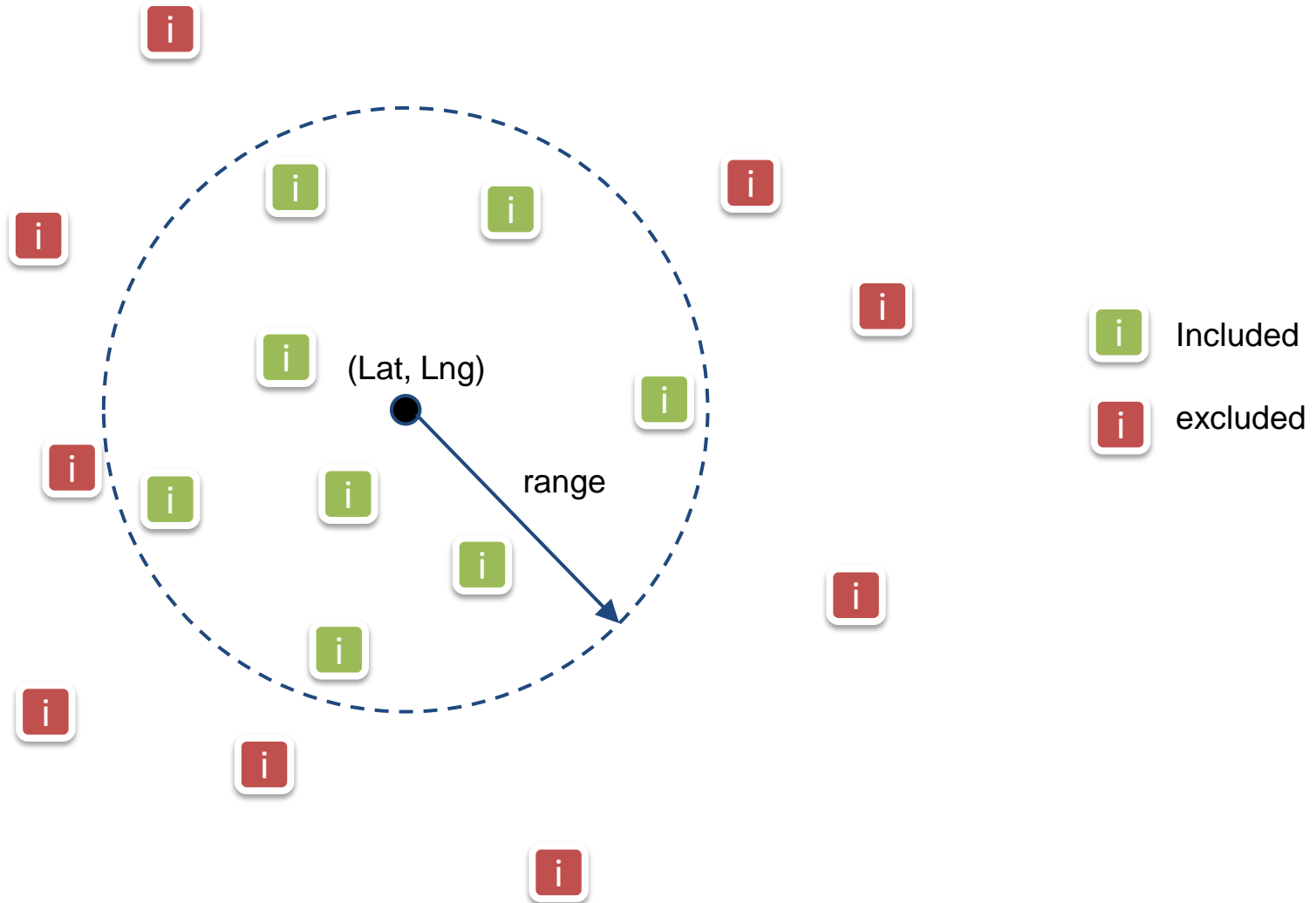
```
<tag >  
  <attachedFileURL></attachedFileURL>  
</tag>
```

# GetTag

- Obtains tags in a vicinity.
- Input :
  - ...
  - lat,lng
  - range
  - queryXML

# GetTag

[parameter : range]



# QueryXML

- XML used to condition the tag query.
  - Use not only in WebALP3::Service::GetTag
- Example :
  - Title of the tag starts with 'TAG', AND
  - registerDate is between '2010-01-01 00:00:00' and '2010-06-30 23:59:59'

```
<?xml version="1.0" encoding="UTF-8"?>
<querySet xmlns="http://www.iwane.com/ALV/">
  <query logical="and">
    <query>
      <targetElement>title</targetElement>
      <like>TAG%</like>
    </query>
    <query>
      <targetElement>registerDate</targetElement>
      <between>2010-01-01 00:00:00,2010-06-30 23:59:59</between>
    </query>
  </query>
</querySet>
```

# GetTagByPaging

- Obtains tags in a particular order.
- Sortable columns
  - “tagKindID”, “tagCategoryID”
  - “groupID”, “groupName”, “groupType”
  - “classID”, “routeID”, “projectID”
  - “tagID”, “id”, “m”, “title”, “keyword”
  - “registerDate”, “updateDate”
- LIMIT, StartingIndex

# DEMO

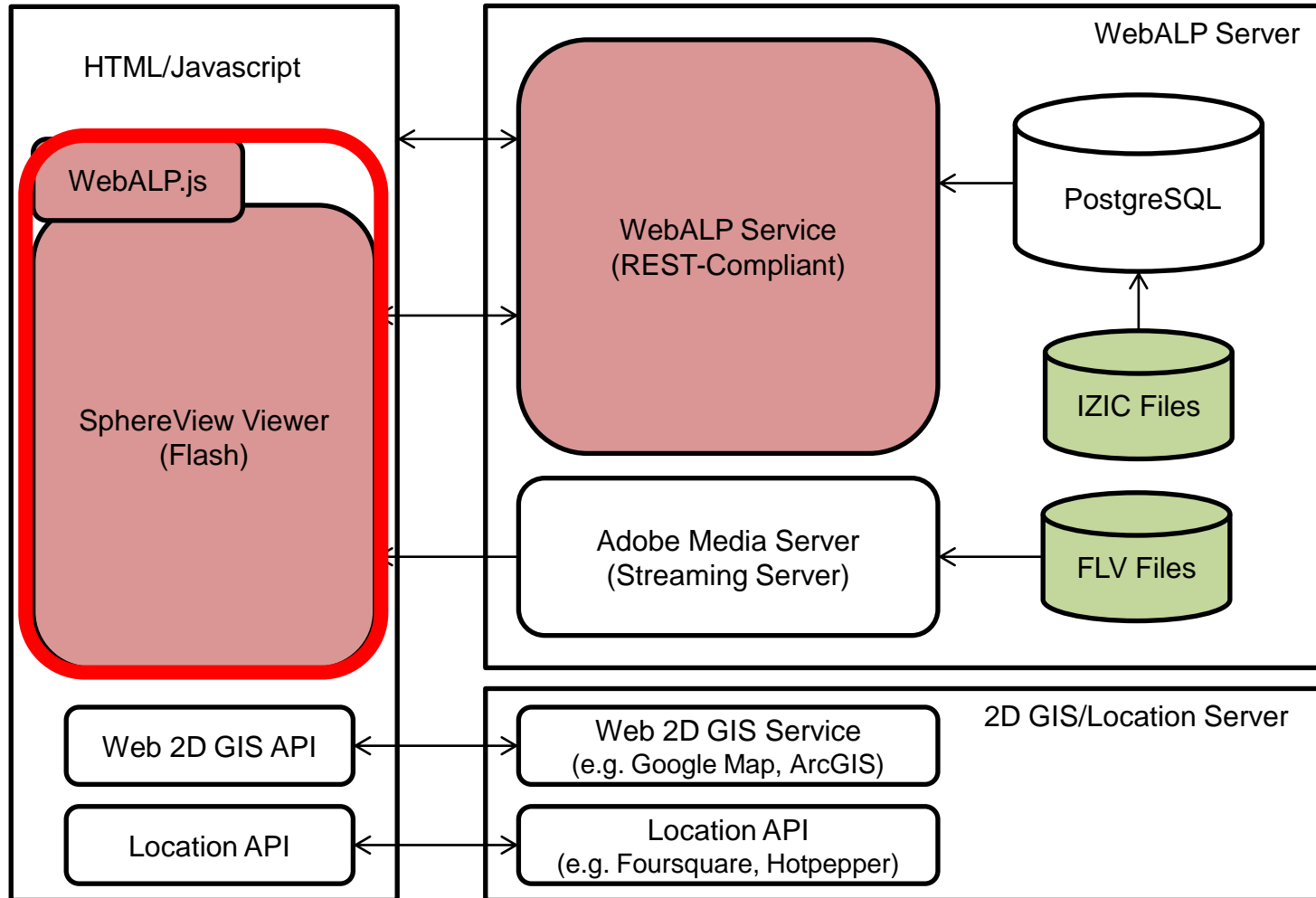
- Check networking debugging
  - Overlay on googlemap sample program
  - TagManagement sample program, DMT

# **SPHEREVIEW AND ITS JAVASCRIPT API**



**Client-side  
(Web Browser)**

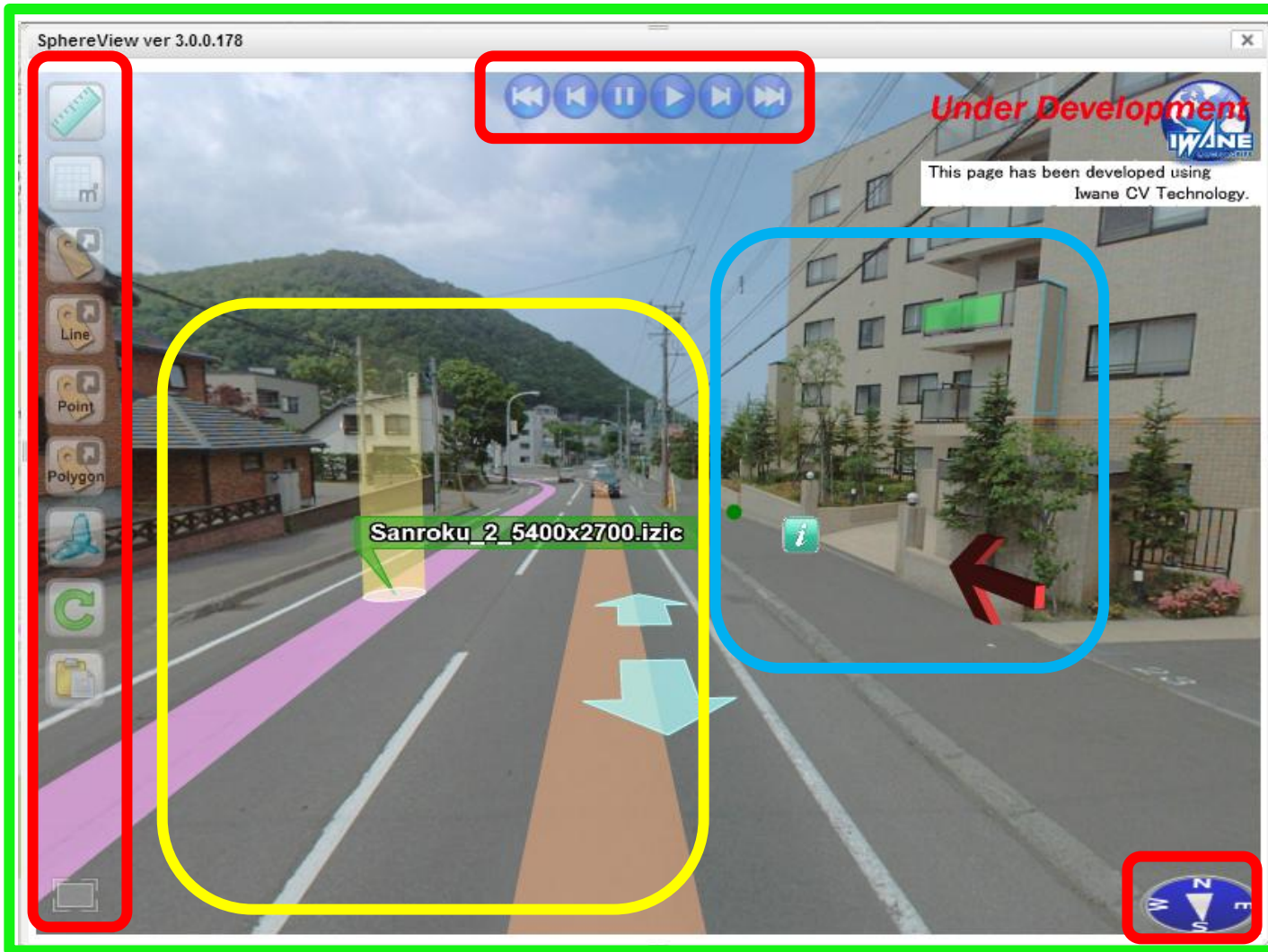
**Server-side**



# Supported Browser

- IE7 and higher
  - Google Chrome
  - Firefox
  - Safari\*
- 
- Conventional browsers should work fine.

# SphereView UI

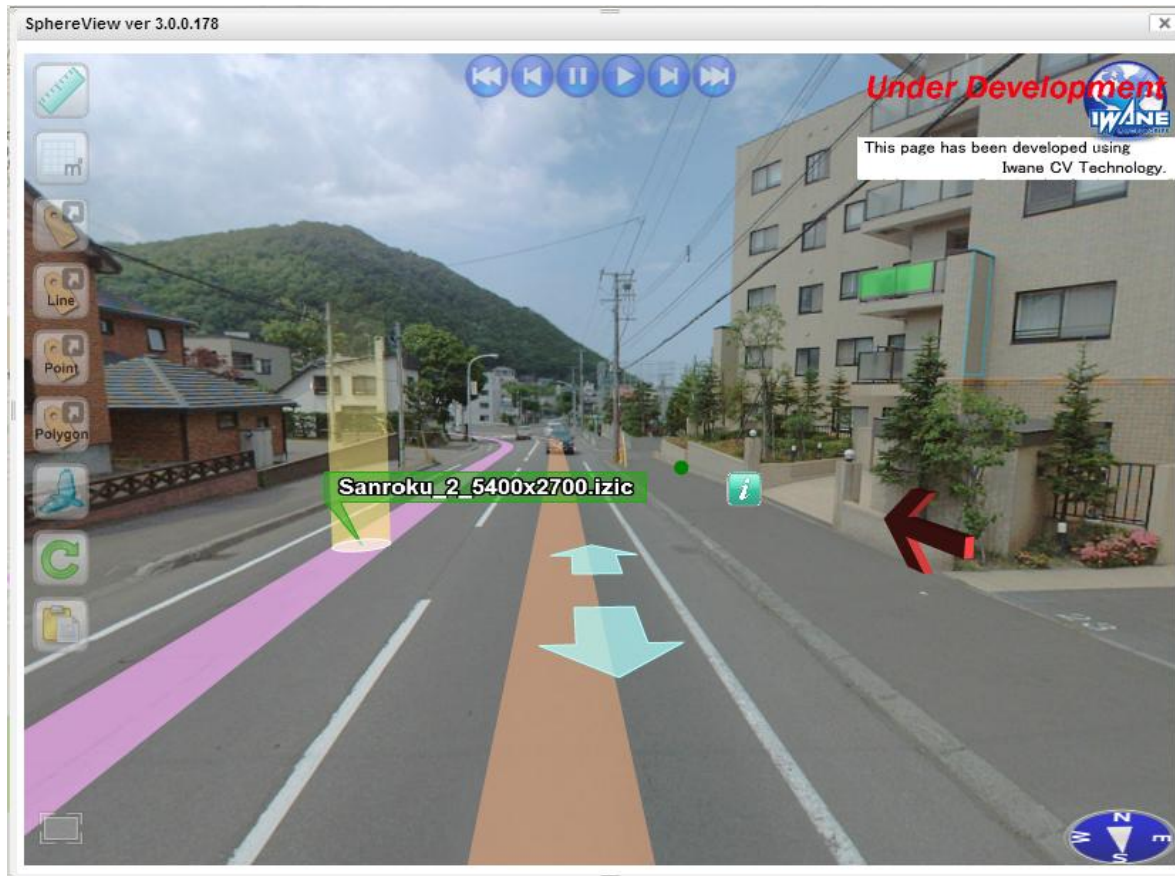


— Base Window  
— Screen Widgets

— Movie Path Layer  
— Tag Service Layer

# Base Window and Screen Widgets

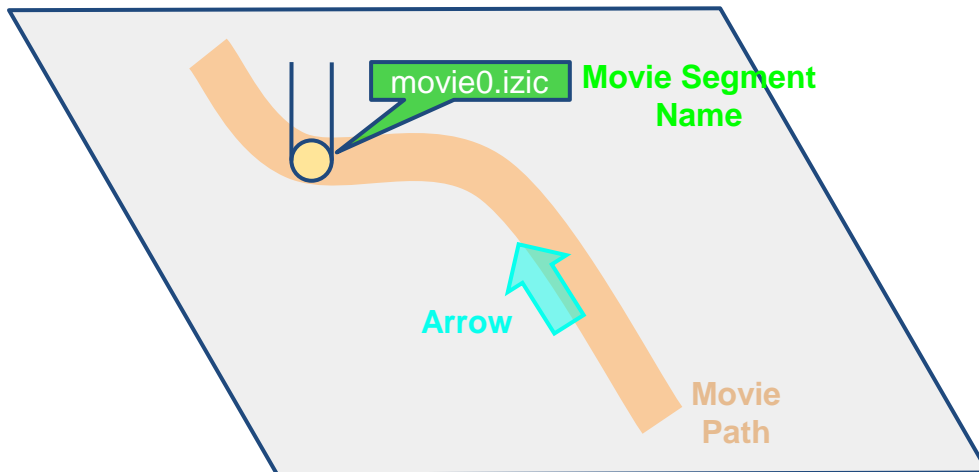
- Demo : Bird's eye view, Measurement, Tag, Streaming



# Movie Path Layer

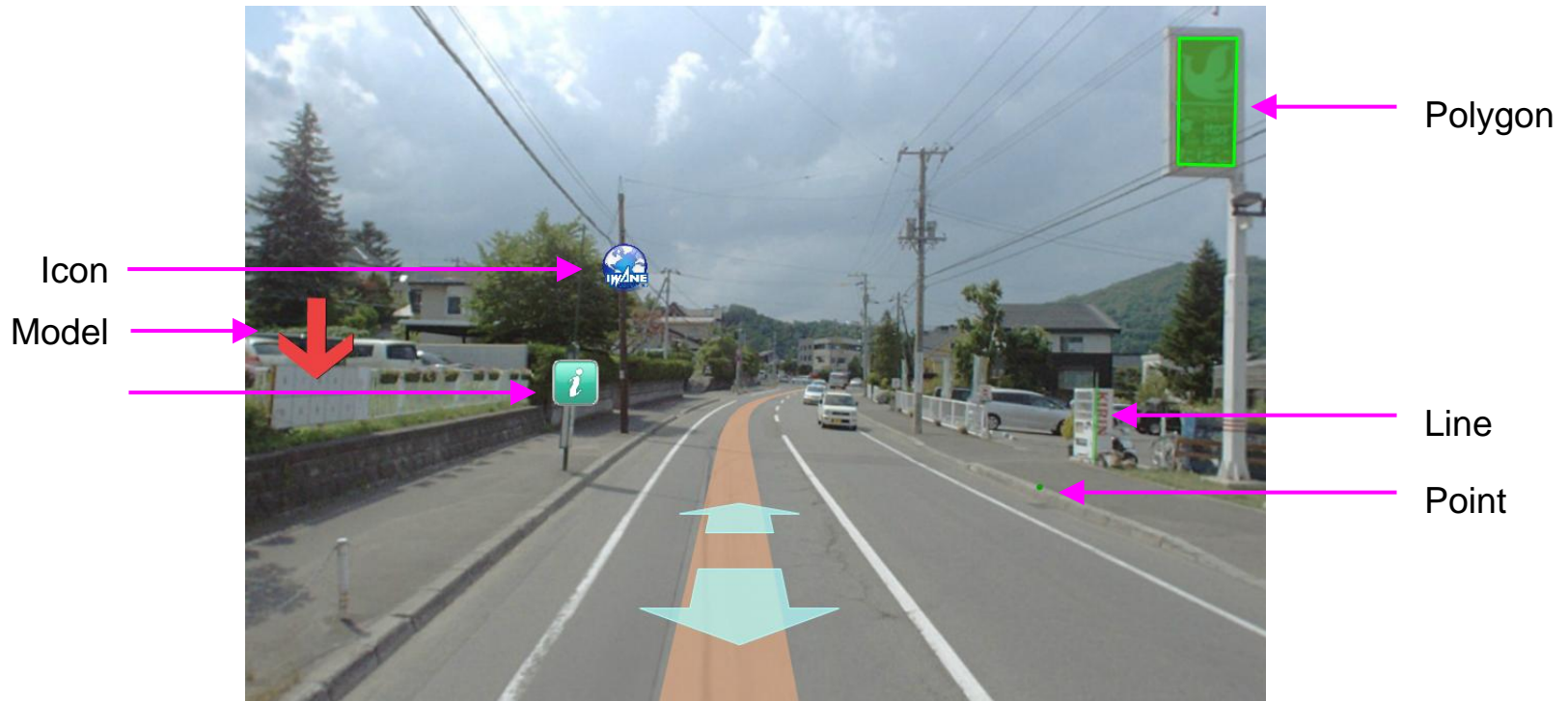
- Main components
  - A path of a movie segment (shorten as movie path).
  - Arrows (play streaming video)
  - A name of a movie segment.
- Existed by default

Layer



# Tag Service Layer

- Display tags in the space.
- Not existed by default.



# SphereView JavaScript API

## [webALP.js]

Namespace : WebALP

SphereView

Callback

Camera

MoviePathLayer

MovieController

TagServiceLayer

Namespace :  
WebALP.TagXML

TagElement etc.

# Class: SphereView and its Callback

- Constructor
  - HTML Document id
  - URL of sphereView.swf
  - URL of expressInstall.swf
  - Callback object
- Login
  - Arrays of Web-Services
- Callback
  - Overrides necessary functions.

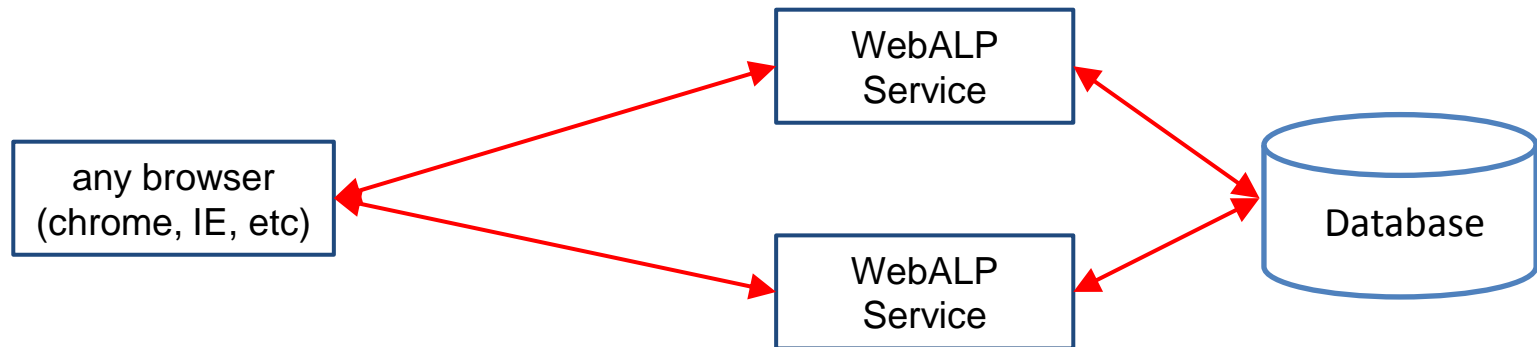


# Let's get our hands dirty

- SphereView Initialization Example.

# Why Array of Web-Services?

- When there is large demand to the web-service, this functionality can help doing a load-balancing.
  - Round-robin manner.
- \*two or more WebALP Service server connecting to the same database



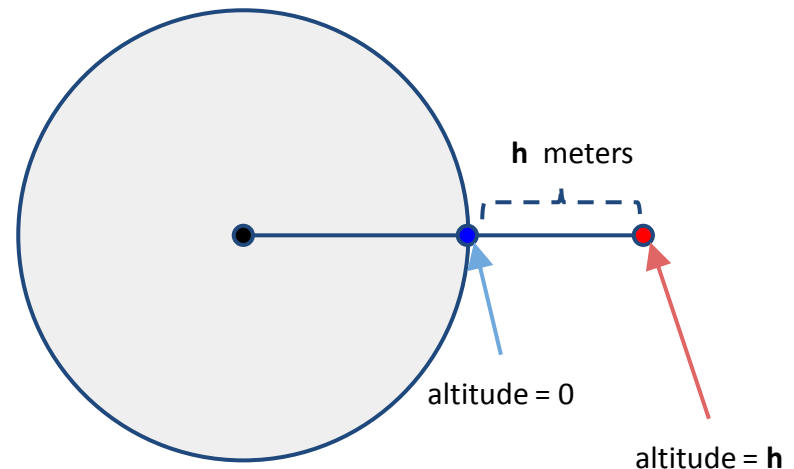
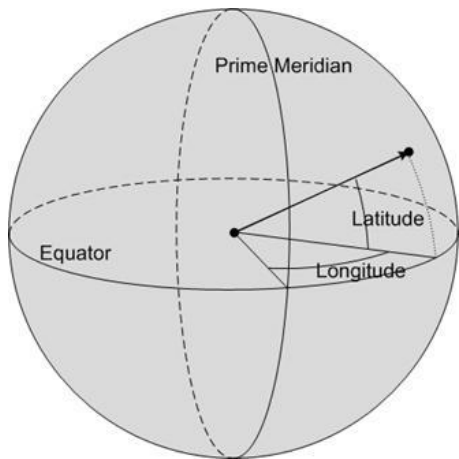
# Class : Camera (viewport)

- Various configuration/behaviors of a camera can be obtained/configured using this class.
- E.g.
  - GetCoordinate()
  - Get/SetFOV()
  - Get/SetBirdEyeView()
  - Get/SetViewingDistance()
  - RequestScreenCapture()

# Coordinate

## Latitude-Longitude-Altitude

- Latitude/Longitude represents a location on a sphere.
- Altitude represents a height from a sea-level (or a zero-height elevation) at a particular Latitude/Longitude.

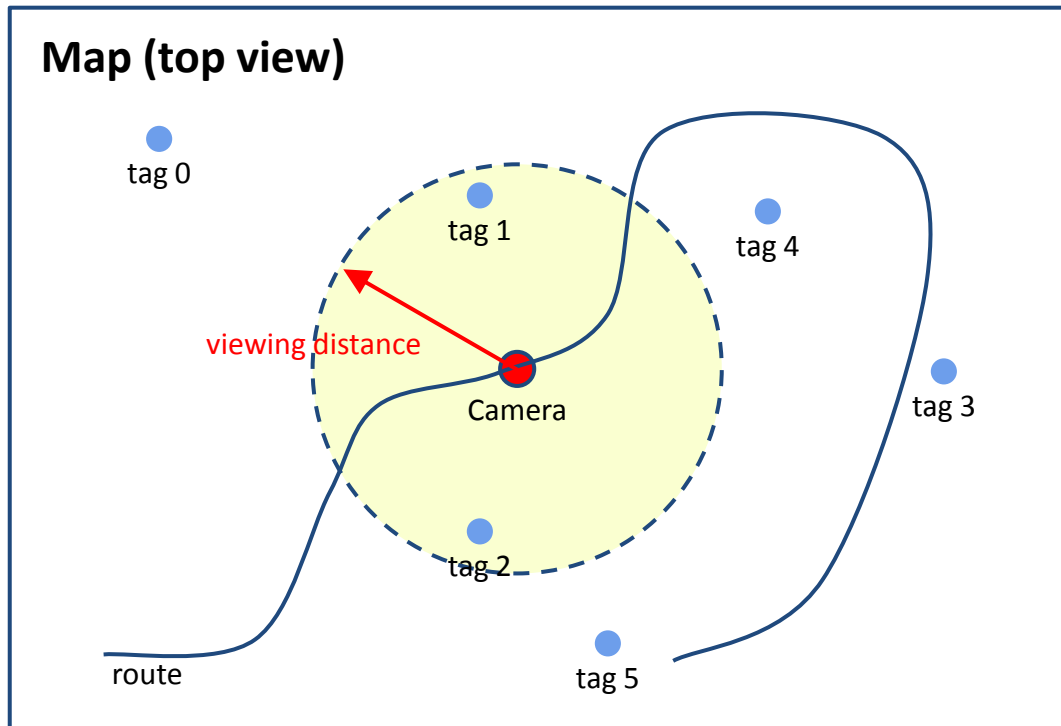


Note : [\[http://technet.microsoft.com/\]](http://technet.microsoft.com/)

- The coordinate (location) information is usually calculated from GPS.
- zero-height elevation may be defined differently, depending on the GPS system used. Please refer to the GPS documentation for further details.

# Viewing Distance

Distance from the center of the camera (in all direction as a circle) that the viewer will show. This is including an information on route, tags, etc.

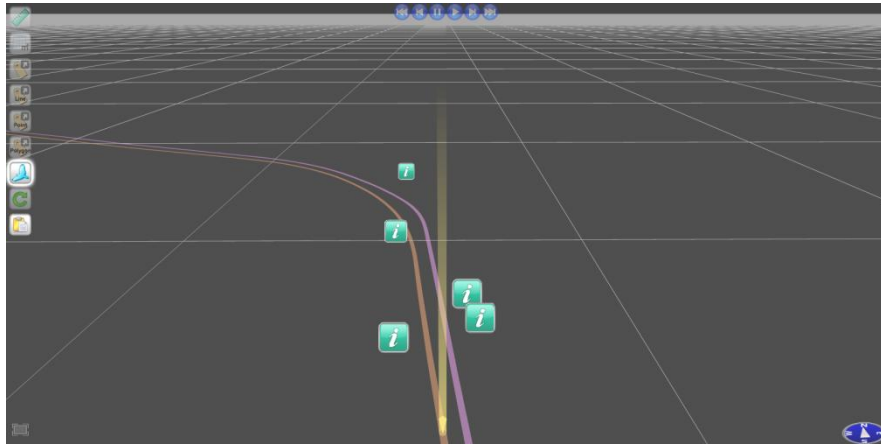


viewing distance = 100 m

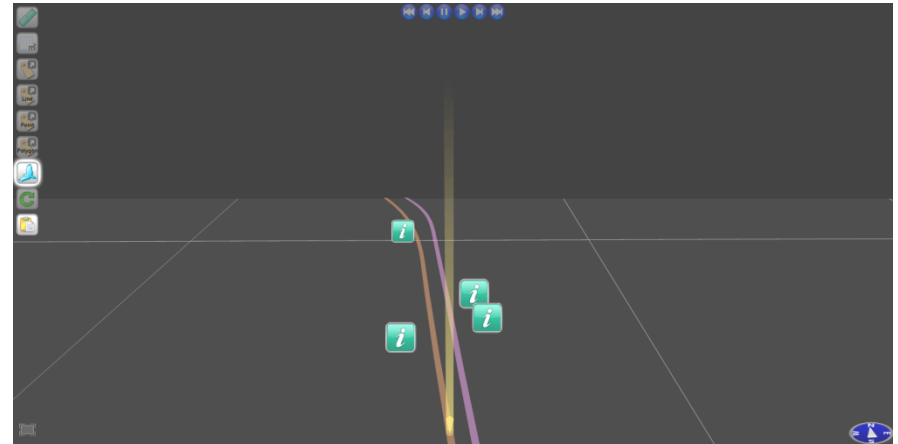


viewing distance = 25 m

# Viewing Distance (Bird Eye View)



viewing distance = 10000 m



viewing distance = 150 m

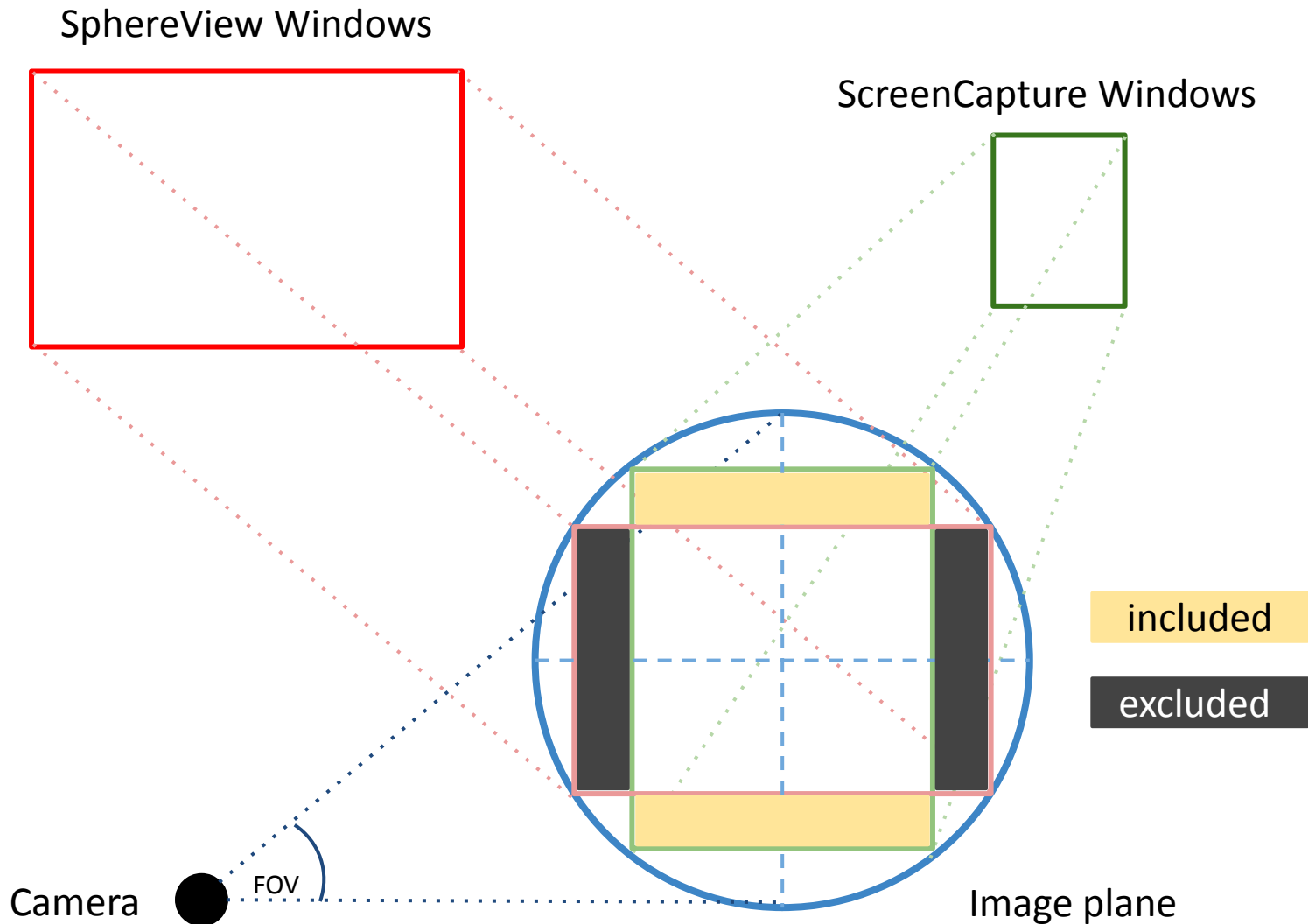
## Note :

- For BirdEyeView, it is a distance from Camera, not from current position (yellow highlight).
- Default values are 10000m:100m [BirdEyeView:NormalView]. This value is not related to the amount of data that will be downloaded.

# Differences between ScreenCapture and ScreenShot (Widget)

- Basically, ScreenShot (Widget) simply copied what you saw on the screen to the system clipboard.
- However, ScreenCapture will
  - Return the best-resolution image based on your viewing angle.
  - MoviePathLayer and TagServiceLayer will (or will not) be rendered, based on their visibility value.
  - When [width, height] are different from the SphereView window, a (slightly) different image is returned. The returned images are calculated by keeping the (diagonal) FOV and image-centre unchanged.

# FOV and Image-center Unchanged





# Try Out?

- Control Camera Example.
- Camera object instance is a SphereView instance property.

# Class : MovieController

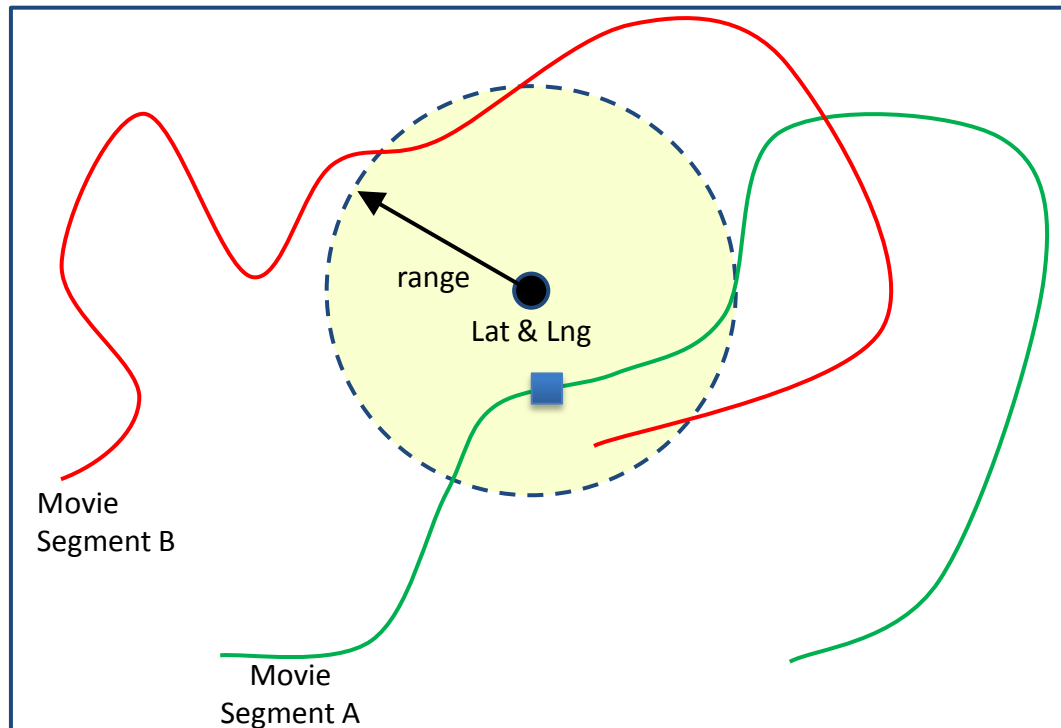
- Configuration regarding movie playback (streaming video) can be obtained/configures from this class.
  - Also location changes of the camera.
- E.g.
  - Get/SetMovieSegmentID
  - Get/SetStreamConfig
  - SeekAtLatLng
  - SeekByDistance

# StreamConfig

- `streamConfig:forwardProjectType = `FORWARDTYPE``
- `streamConfig:reverseProjectType = `REVERSETYPE``
- `streamConfig:suffix = `SUFFIX``
- `streamConfig:streamFileType = `FILETYPE``
- For example, Sapporo0001.izic
  - Template Stream Name = ``STREAMTEMPLATE``
- Forward play button; `Play()`
  - `lowercase(`STREAMTEMPLATE`_`FORWARDTYPE`_forward_`SUFFIX`_.`FILETYPE`)`
- Reverse play button; `ReversePlay()`
  - `lowercase(`STREAMTEMPLATE`_`REVERSETYPE`_reverse_`SUFFIX`_.`FILETYPE`)`

# SeekAtLatLng

- Search for a frame closest to a location provided by Latitude&Longitude, within a range condition.

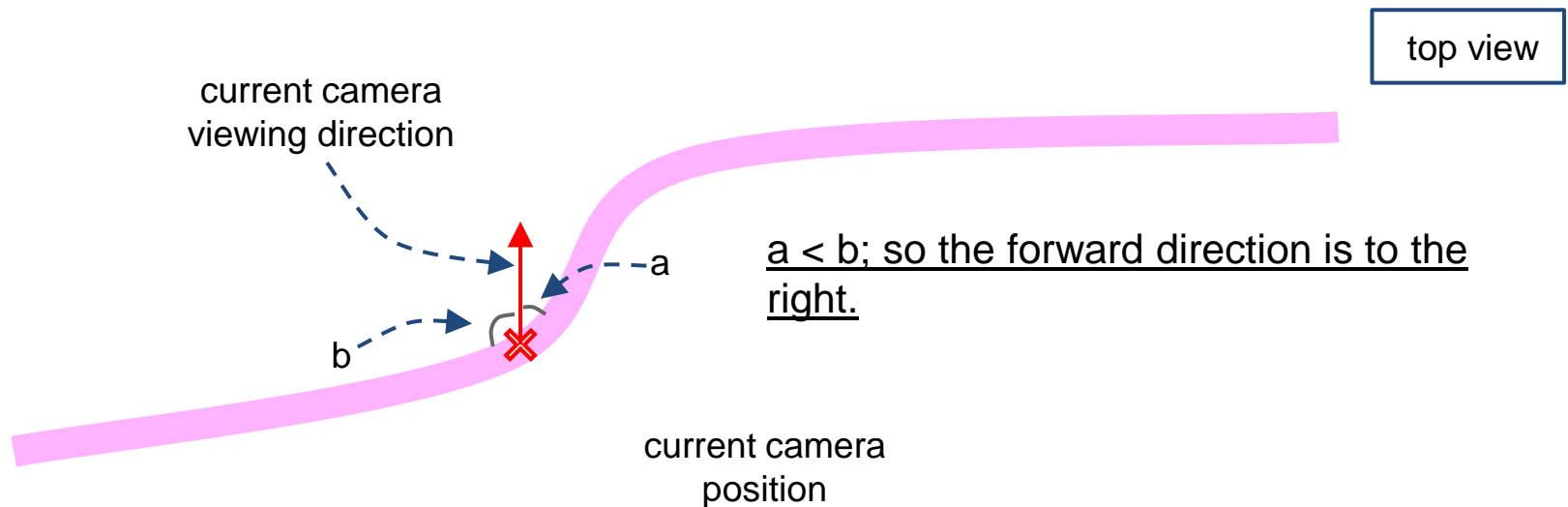


# SeekByDistance

Search for a frame that is separated from a current frame by a given distance along the trajectory. Once found, camera will move to that frame.

The search is only conducted on a 'forward' direction of the current movie path ONLY. The 'forward' direction is decided by the current viewing direction, e.g.

The current movie path  $\rightarrow$  same m. segment, or within connect m. segment.



# Default GroupID of MovieController

- By setting a default GroupID, only videos in these groups will be searched during ANY seek operation.
  - Separating group of data; e.g. road and path way.
  - for speed
- If blank array is set, every group in the database will be considered.

# Back to Coding Again.

- MovieController Seek Example.
- MovieController object instance is a SphereView instance property.

# Class MoviePathLayer

- Configuration regarding movie playback (streaming video) can be obtained/configures from this class.
  - Also location changes of the camera.
- E.g.
  - Get/SetHeightFixedMode()
  - Get/SetRange()

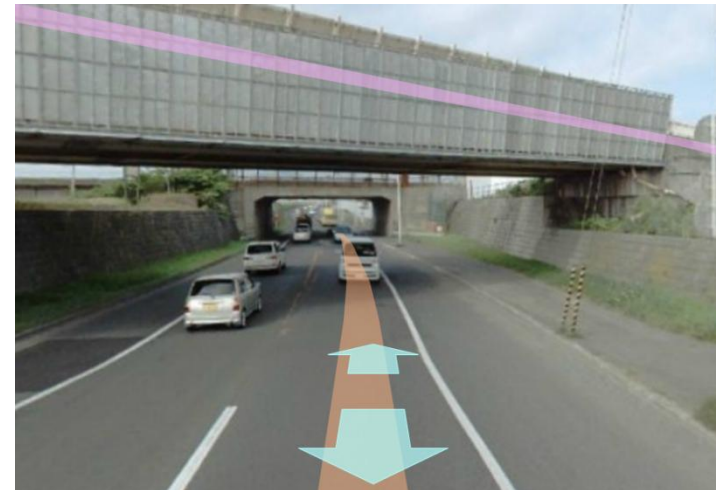
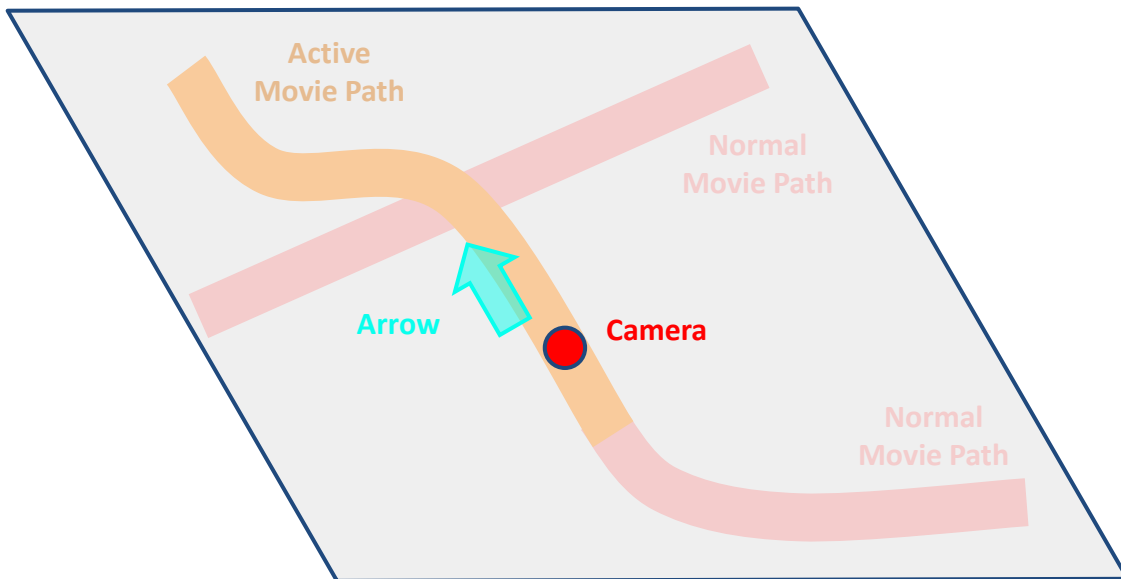


# Movie Path

There is two types of movie path.

1. Normal (default) movie path. This includes ALL available movie paths, except the active one.
2. Active movie path. This refers to the move path that the camera is following.

Layer (Normal sphere view)



# MoviePath Range

- Distance around the camera (in all direction as a circle), that movie path will be fetched at once.
  - If only a portion of movie path is in the range, the whole path is fetched.

## Note:

The more the value of range, the slower the sphereview.swf would be. This is because it takes time to fetch and store more data.

# Difference between Viewing Distance & Get Range

ViewingDistance limits the area where the information (Movie Path, Tags, etc.) will be shown to the user.

GetRange describes the area where the information will be fetched by the sphereview.swf

This means that even if the data are fetched, but viewing distance is shorter, the information will not be shown.

# Back to Coding.

- Movie Path Layer Example.
- MoviePathLayer object instance is a SphereView instance property.

# Class TagServiceLayer

- Configuration regarding how tag should be displayed.
- E.g.
  - Get/SetQuery
  - GetAutoUpdate/SetAutoUpdate
  - Get/SetRange

# Getting TagServiceLayer Instance

- SphereView.AddTagServiceLayer
  - layerName (unique)
  - serviceURL
  - Range
  - ...
  - queryXML
- SphereView.GetTagServiceLayer

# TagServiceLayer and GetTag

- SphereView makes use of WebService GetTag method.
- Custom Tag WebService?
  - Make sure Input parameter and returned value is the same as WebService GetTag.
  - Specify serviceURL when adding the layer to SphereView.

# Get/SetQuery

- Equivalent to QueryXML parameter of WebService.GetTag
  - XML is the same.



# AutoUpdate

- Whether SphereView should automatically call `WebService.GetTag` in a specific time interval.
  - `UpdateInterval`
- If not, then update can be manually called from JavaScript as well.
- Unsync between `SphereView.GetTag` & `Webservice.GetTag`

# Range

- Similar to Range in `WebService.GetTag`
- Why it is so important?
  - SphereView VS Displaying on google map.
  - SphereView VS SphereView's bird eye view.

# ProxyTagServiceLayer

- Don't like custom tag webservice.
- Want to do mash up from other GIS service.
  - E.g. facebook, foursquare
- Fetch data from their webservice , convert to TagXML format, and display in a SphereView.
  - Can all be done through JavaScript.

# Let's Get our hands Dirty again!

- Tag Service Layer example
- Proxy Tag Service Layer example

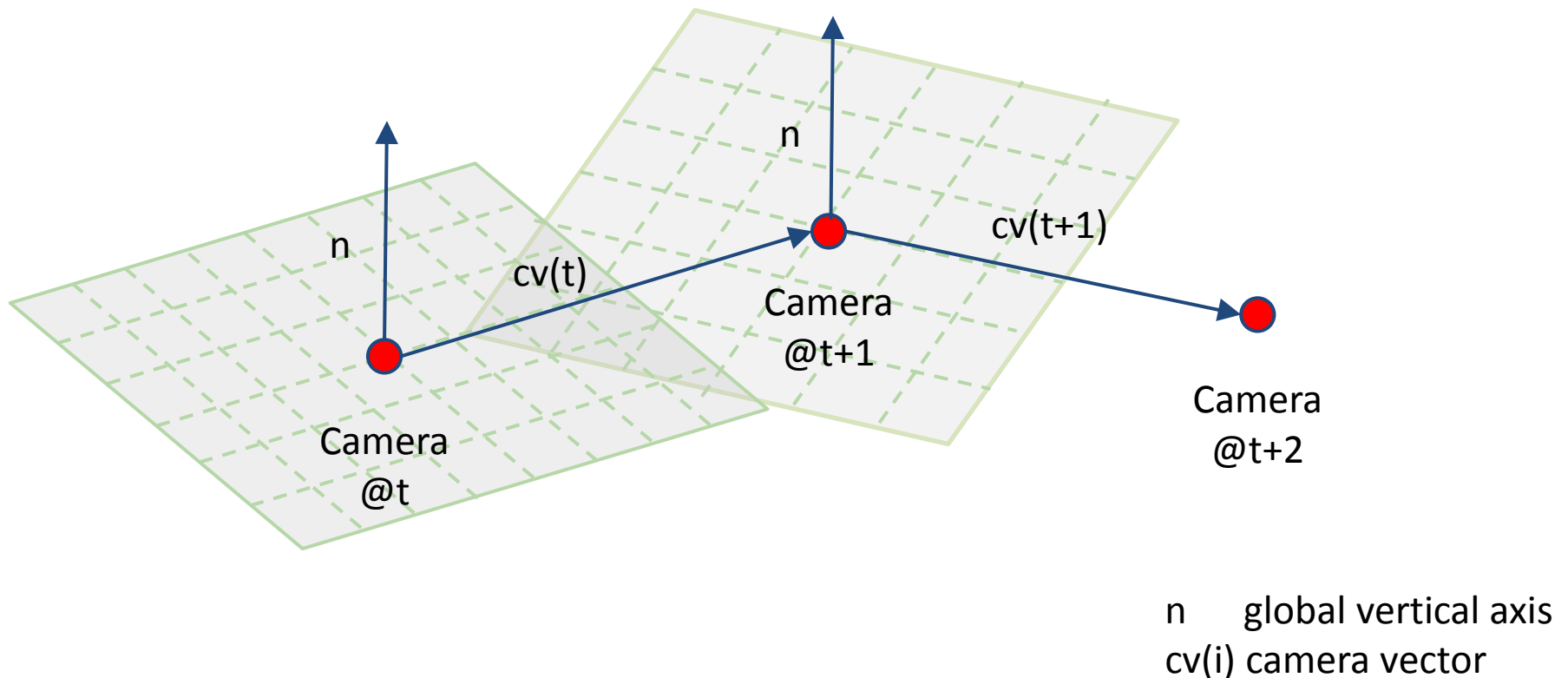


# Measurement Methods

1. Horizontal grid
  2. Cube grid
  3. Epipolar
  4. Measurement 3D
- Can be change using
    - Get/SetMeasuringMethod

# HORIZONTAL\_GRID

From the calculated camera vector, a horizontal plane is created to help user to choose measurement points.

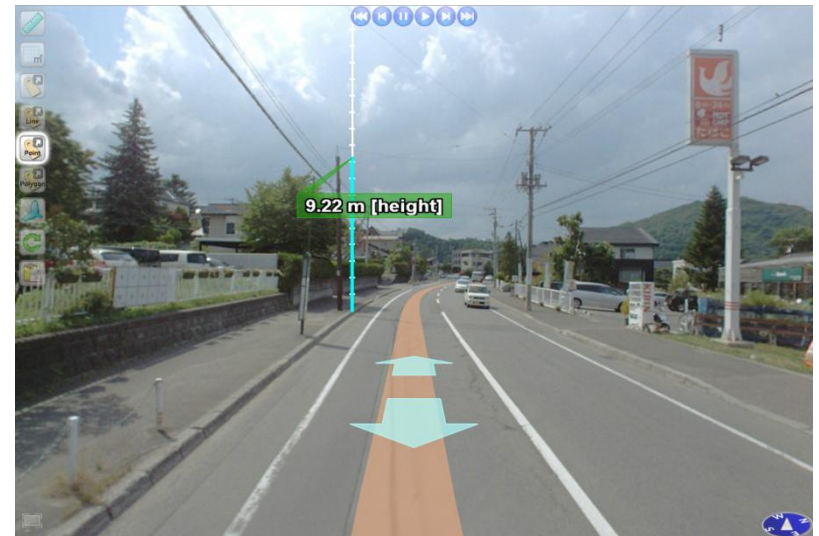


# HORIZONTAL\_GRID ... Steps

1.) Choose a position on the grid.



2.) Choose a height.

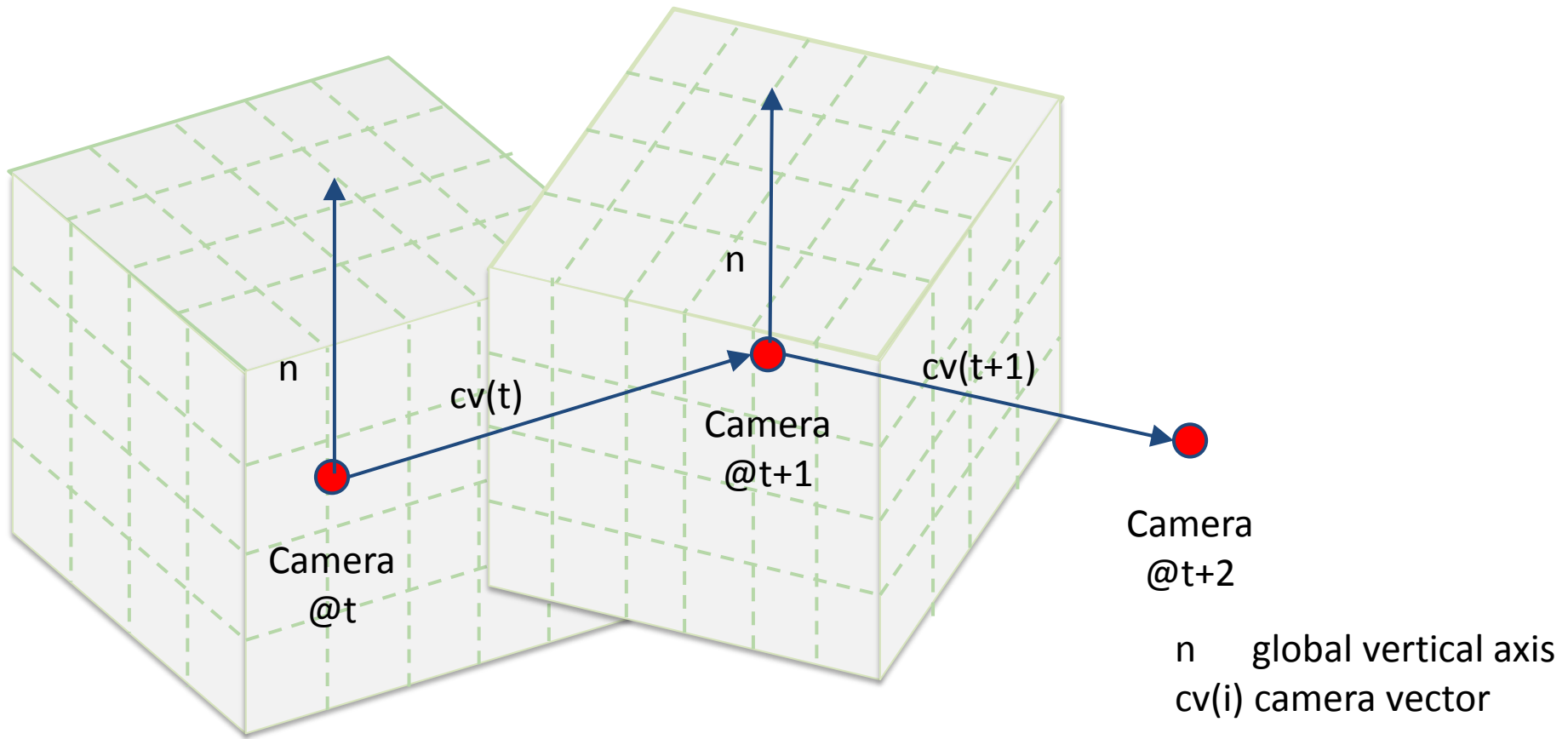




# CUBE\_GRID

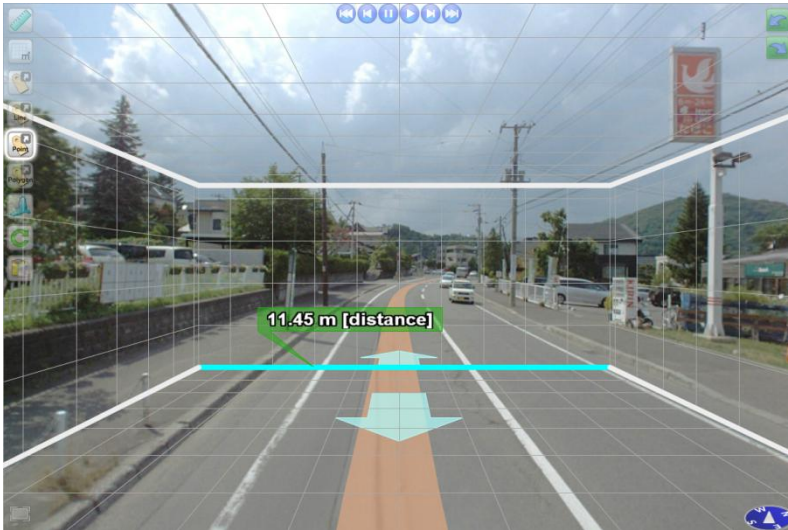
From the calculated camera vector, a cube grid is created to help user to choose measurement points.

- \*Only points on the GRID will be assigned.

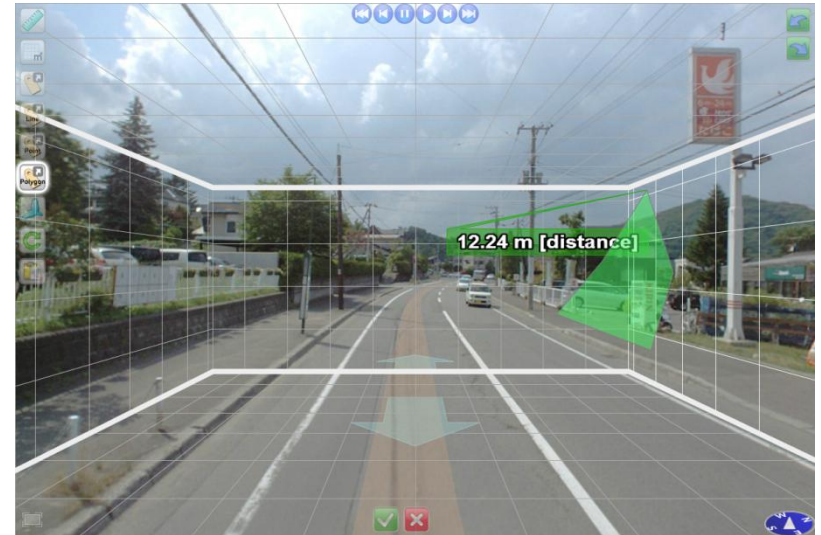


# CUBE\_GRID ... Steps

1.) Change grid's size and orientation

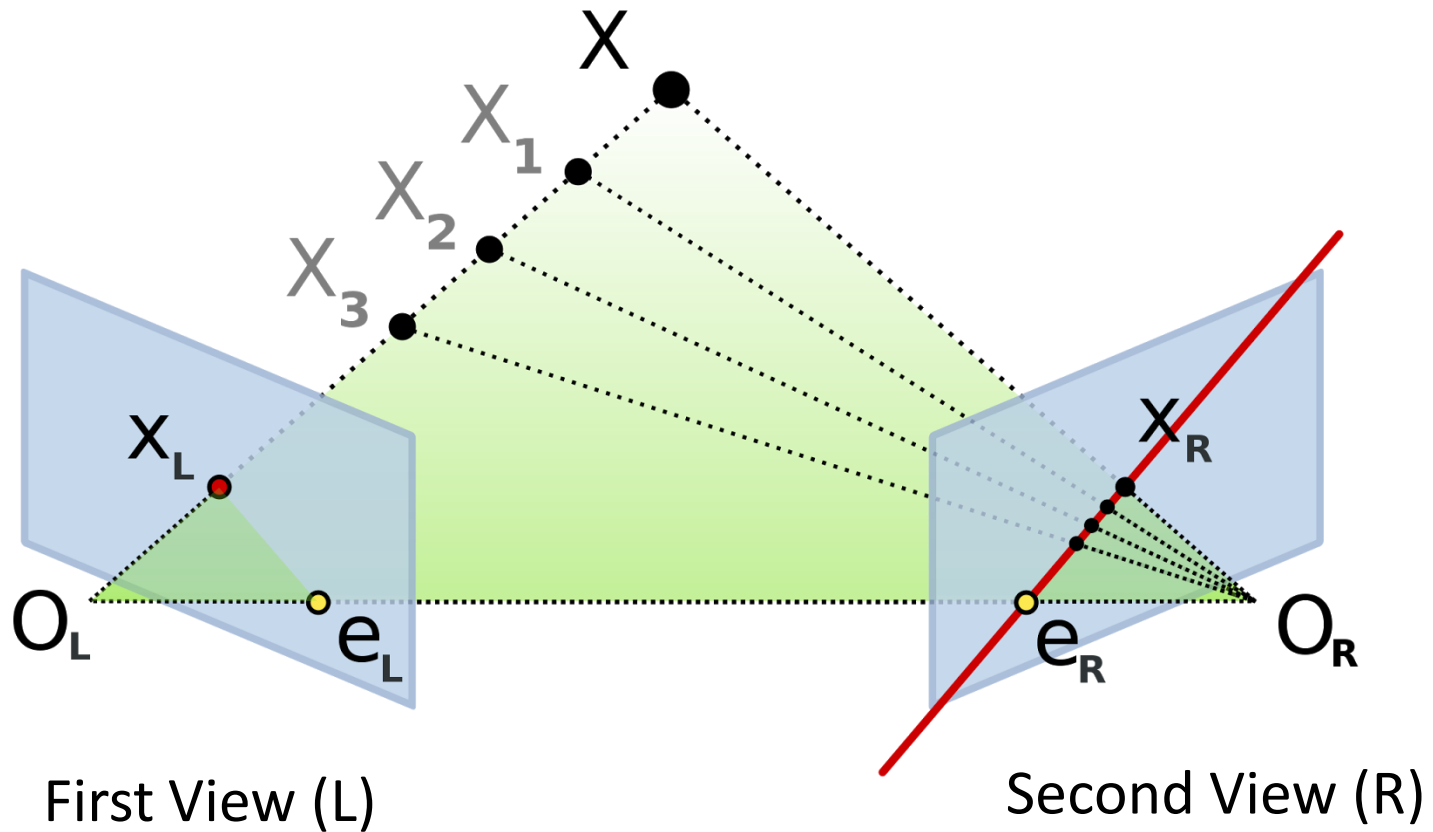


2.) Choose points on the grid.



- Grid can be rotated, parallel to the horizontal plane.
  - Click on top-right buttons.

# Epipolar Geometry



# EPIPOLAR ... Steps

1.) Choose a point in the first image.

2.) Choose its corresponded point on the red line from the second image.



- Once a point is clicked in the first image (or a upper camera), a camera view will change automatically to the bottom camera view.

# MEASUREMENT\_3D (AUTO)

Based on the epipolar geometry, once a user choose a point on an image, the system tries to find its 3D coordinate automatically.



# Let' try it together.

- Measuring Method example

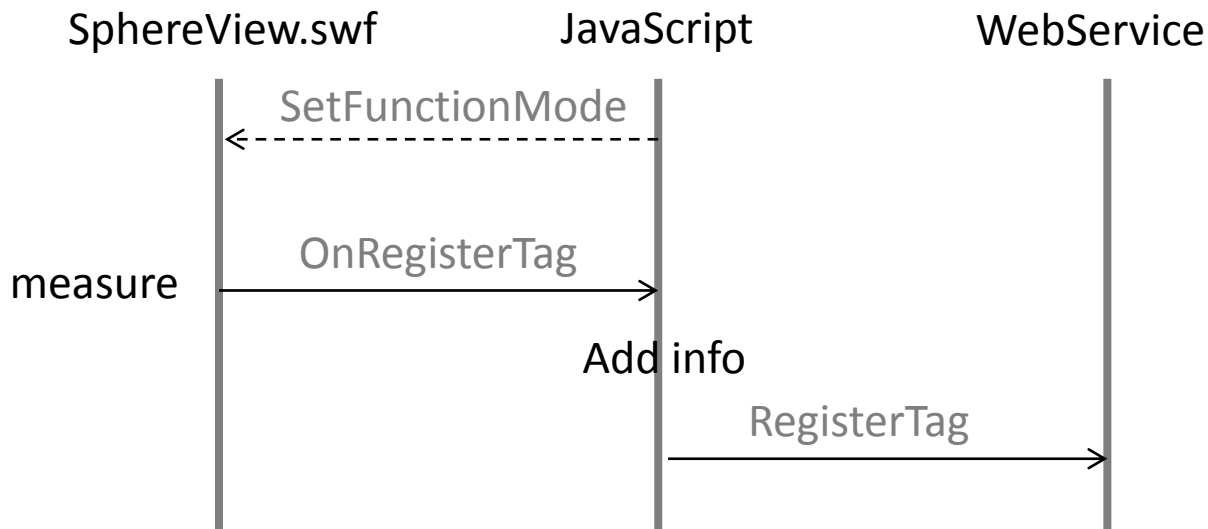
# Adding/Editing Tag

- WebALP Service
  - Registering information to the database
  - WebALP.RegisterTag
- SphereView API
  - Measuring 3D information, and return value to JavaScript
  - WebALP.SphereView.Callback.OnRegisterTag
  - WebALP.SphereView.Callback.OnEditTag

# Adding Tag

## [Icon, Point, Line, Polygon]

- Entering REGISTER\_\*\_TAG mode
- Measure 3D information
- Receive information through Callback.OnRegisterTag()
- Add necessary information
- Adding tag to database WebALP.RegisterTag





# Editing Tag

## [Icon, Point, Line, Polygon]

- Getting an access to the tag
  - `WebService.GetTag` or `SphereView.Callback.OnSelectTag`
- Modifying tag's information
  - Directly edit XML.
  - OR use `WebALP.TagXML.TagElement` utilities.
- Updating tag to database `WebALP.RegisterTag`
  
- Note
  - 3D information CANNOT be re-measured in `SphereView`.

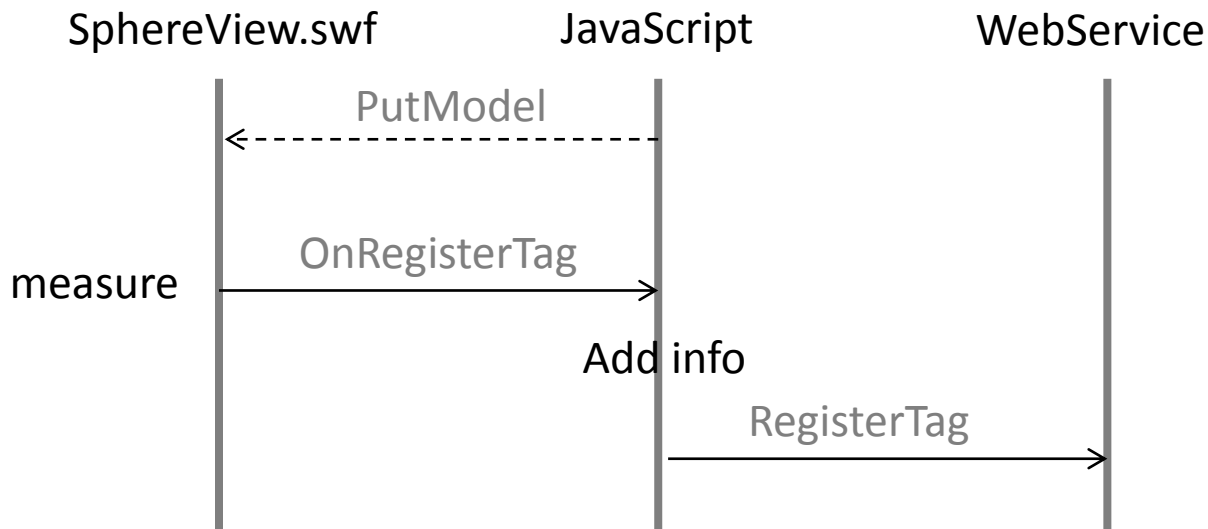
# GetTag

- `WebService.GetTag` VS `TagServiceLayer.GetTag`
- In general, they should contain the same infos.
- However, if database has been update but the layer hasn't updated yet, they might not be the same.
  - This is the same for `Callback.OnSelectTag`

# Adding Tag

## [Model - Collada]

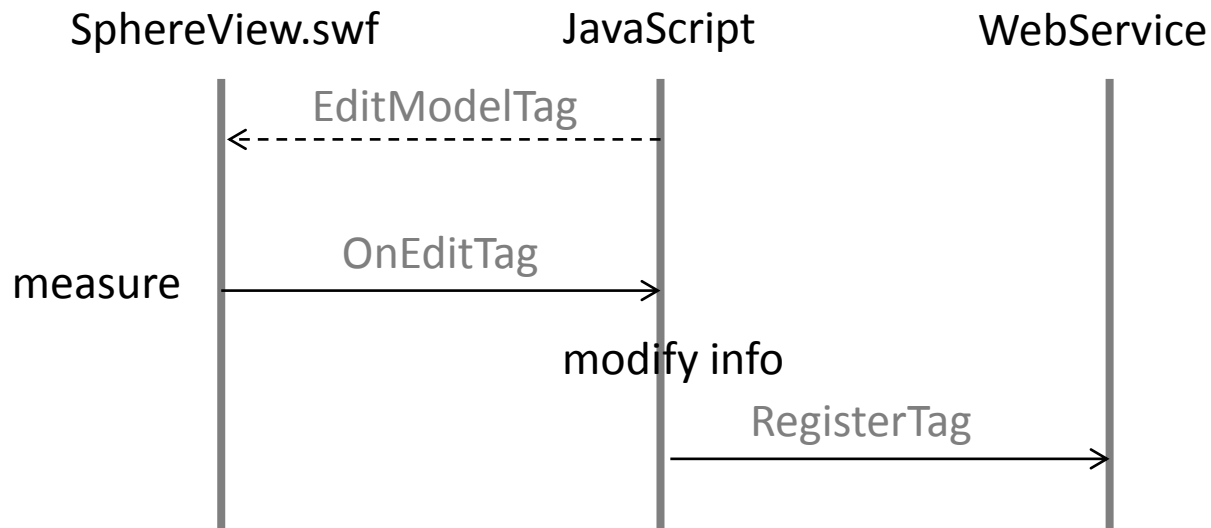
- SphereView.PutModel with model's URL
- Measure 3D information, and aligned model
- Receive information through Callback.OnRegisterTag()
- Add necessary information
- Adding tag to database WebALP.RegisterTag



# Editing Tag

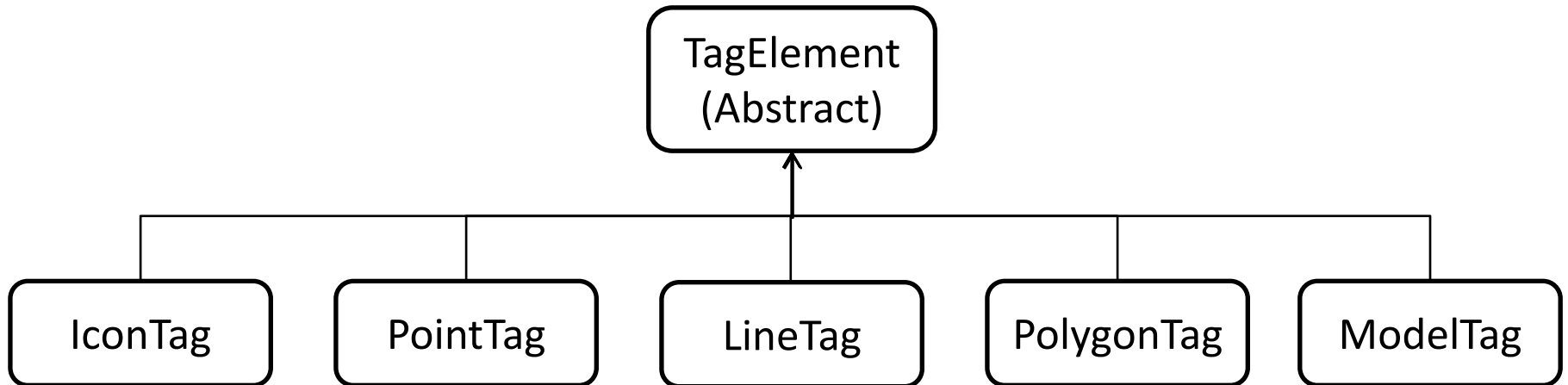
## [Model - Collada]

- Getting an access to the tag
- SphereView.EditModelTag with tag's id
- Modify 3D information, and aligned model
- Receive information through Callback.OnEditTag()
- Modifying tag's information
- Updating tag to database WebALP.RegisterTag



# WebALP.TagXML

- Utility class to modify XML of a tag.



- `TagElementFactory.CreateFromXMLString`
- `TagSetFactory.CreateFromTagElementArray`

# TagElement Example

```
<tag xmlns="http://www.iwane.com/ALV/" id="2622" >
  <title>TAG_POINT</title>
  <point>
    <altitudeType>cvDataCoordinate</altitudeType>
    <coordinates>43.0394980385414,141.329988275941,-54.5565716710095</coordinates>
  </point>
  ...
</tag>
```

- Get
  - tagElement.id(), tagElement.title()
- Set
  - tagElement.id("3000"), tagElement.title("new\_title")

# Tag's Icon (IconTag Only)

```
<tag >
  ...
  <icon>
    <URI></URI>
  </icon>
  <tagKind>
    ...
    <icon>
      <URI></URI>
    </icon>
  </tagKind>
</tag>
```

- Priority
  1. Tag->icon->URI
  2. Tag->tagKind->icon->URI
  3. noIconURL of SphereView.AddTagServiceLayer
  4. Default Tag Icon (green icon)

# Cross Domain Issue

- Where SphereView client wants to access data across domains using a HTTP protocol, a destination domain must permits an access.
  - crossdomain.xml
- Situations
  - SphereView wants to display image of a Icon
  - SphereView wants to display Model (collada file)
- Reference : <http://www.adobe.com/devnet/adobe-media-server/articles/cross-domain-xml-for-streaming.html>



# Last Coding Session

- TagManagement sample
- Let's see all sample codes in the gallery as well.

# **QUESTION AND ANSWER**

# Further Inquiries

- E-mail
  - [technicalsupport@iwane.com](mailto:technicalsupport@iwane.com)
- Forum
  - <http://iwanelab.sakura.ne.jp/forum/index.php>

# **APPENDIX**