AVSUM EVENTS

AVSUM supports ONVIF Event Service to receive event notifications from the ONVIF Event Server in real-time. This service is implemented in the C# as follows:

1. Add Event Service WSDL as a web reference to obtain ***EventService*** namespace. Event Service WSDL is located at the following link: http://www.onvif.org/onvif/ver10/event/wsdl/event.wsdl

2. Create ***EventService.EventBinding*** object and set it's URL:

```
EventBinding evBind = new EventBinding();
evBind.Url = URL;
```

   URL is the address of the Event Server.

3. Then call ***CreatePullPointSubscription*** function. This function returns ***EventService.EndpointReferenceType*** object which contains URL to the Event Service for the current client:

```
FilterType filter = new FilterType();
string initTermTime = "";
CreatePullPointSubscriptionSubscriptionPolicy policy =
   new CreatePullPointSubscriptionSubscriptionPolicy();
XmlElement[] elems1 = new XmlElement[10];
DateTime  time1 = new DateTime();
Nullable<DateTime>  time2 = new DateTime();

EndpointReferenceType endPoint =
   evBind.CreatePullPointSubscription(filter, initTermTime,
   policy, ref elems1, out time1, out time2);
```

   Meanings of all the parameters are described in the Event Service WSDL available at the link given above.

4. If PullPointSubscription creation succeeded create ***EventService.PullPointSubscriptionBinding*** object and set it's URL to the returned ***endPoint*** value:

```
PullPointSubscriptionBinding subBind = new
   PullPointSubscriptionBinding();
subBind.Url = endPoint.Address.Value;
```

5. Now we are ready to receive events. This is done in a separate thread because event obtaining is implemented synchronously. To pull events a function ***PullMessages*** is used:

```
string timeOut = "PT60.000S";
int mesLimit = 10;
XmlElement[] elems2 = new XmlElement[10];
DateTime termTime = new DateTime();
NotificationMessageHolderType[] messHolder =
   new NotificationMessageHolderType[0];

subBind.PullMessages(timeOut, mesLimit, elems2,
   out termTime, out messHolder);

for (int i = 0; i < messHolder.Length; i++)
{
     /* Message parsing */
     ...
}
```

This function returns an array of XMLs as an out parameter. Elements from this array can be parsed as desired. Meanings of all the parameters are also described in the Event Service WSDL available at the link given above. Most important from them are ***timeOut*** and ***mesLimit***. Parameter ***timeOut*** means a maximum time to wait for events after which function returns. Parameter ***mesLimit*** indicates how many XML elements (events) you want to obtain at a time. If more than ***mesLimit*** events are available at the Event Server function returns only first ***mesLimit*** events. The fate of the remaining events is left up to the Event Server implementation. In our case they are just obtained during the next iteration.

6. Upon the end it's necessary to unsubscribe from the Event Service. Create ***EventService.SubscriptionManagerBinding*** object, set it's URL and call function ***Unsubscribe()***. Then dispose all objects that were created. This is done simply as follows:

```
SubscriptionManagerBinding smb = new
   SubscriptionManagerBinding();
smb.Url = endPoint.Address.Value;
smb.Unsubscribe(new Unsubscribe());

subBind.Dispose();
evBind.Dispose();
smb.Dispose();
```