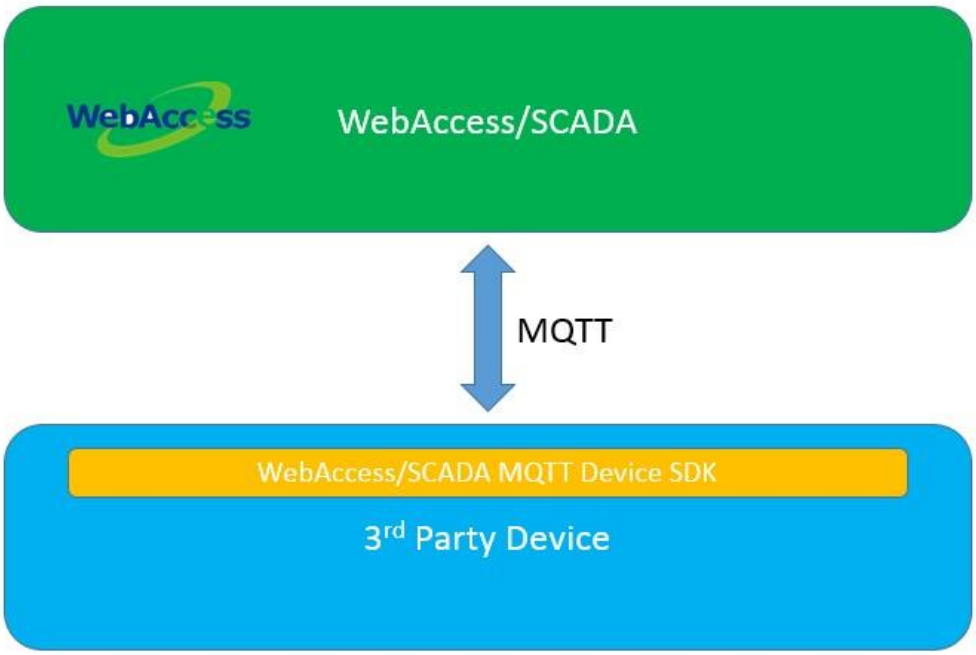
MQTT Device SDK

3rd party device publish data to WebAccess/SCADA

3rd party device receive control message from WebAccess/SCADA



MQTT Device C# SDK

1. Source Code Reference

- Github Repo: MQTT-SDK-for-CSharp
- Service SDK: MQTT.Device.DotNet.SDK
- Sample Code: MQTT.Device.DotNet.SDK.Test

2. Environment

- IDE
  - Visual Studio 2013 or above
- Runtime
  - .Net Framework 4.5.2 or above

3. Instructions

3.1. Constructor(EdgeAgentOptions options)

New a EdgeAgent object.

```
EdgeAgentOptions options = new EdgeAgentOptions()
{
    AutoReconnect = true,
    ReconnectInterval = 1000,
```

```
    ScadaId = textBoxGroupId.Text,
    Heartbeat = 60000,    // default is 60 seconds,
    DataRecover = true,
    ConnectType = ConnectType.MQTT,
    UseSecure = checkBoxSSL.Checked
};
options.MQTT = new MQTTOptions()
{
    HostName = "127.0.0.1", // your broker location
    Port = 1883,
    Username = "admin",
    Password = "admin",
    ProtocolType = Protocol.TCP
};
_edgeAgent.Connect();
```

3.2. Event

EdgeAgent has three event for subscribing.

- Connected: When EdgeAgent is connected to cloud (WebAccess MQTT Broker).

- Disconnected: When EdgeAgent is disconnected to cloud.

- MessageReceived: When EdgeAgent received MQTT message from cloud. The message type as follows:

- WriteValue: Change tag value from cloud.

- WriteConfig: Change config from cloud.

- TimeSync: Returns the current time from cloud.

- ConfigAck: The response of uploading config from edge to cloud.

```
_edgeAgent.Connected += _edgeAgent_Connected;
_edgeAgent.Disconnected += _edgeAgent_Disconnected;
_edgeAgent.MessageReceived += _edgeAgent_MessageReceived;
private void edgeAgent_Connected( object sender,
EdgeAgentConnectedEventArgs e )
{
   // Connected
   Console.WriteLine( "Connect success !" );
}
private void edgeAgent_Disconnected( object sender,
DisconnectedEventArgs e )
{
   // Disconnected
   Console.WriteLine( "Disconnected..." );
}
private void edgeAgent_MessageReceived( object sender,
MessageReceivedEventArgs e )
{
   switch ( e.Type )
   {
```

```
        case MessageType.WriteValue:
            WriteValueCommand wvcMsg = ( WriteValueCommand )
e.Message;
            // wvcMsg content
            break;
        case MessageType.WriteConfig:
            break
        case MessageType.TimeSync:
            TimeSyncCommand tscMsg = ( TimeSyncCommand ) e.Message;
            Console.WriteLine( "UTC Time: {0}",
tscMsg.UTCTime.ToString() );
            break;
        case MessageType.ConfigAck:
            ConfigAck cfgAckMsg = ( ConfigAck ) e.Message;
            Console.WriteLine( "Upload Config Result: {0}",
cfgAckMsg.Result.ToString() );
            break;
    }
}
```

3.3. Connect()

Connect to cloud (WebAccess MQTT Broker). When connect success, the connected event will be triggered.

```
edgeAgent.Connect();
```

3.4. Disconnect()

Disonnect to cloud (WebAccess MQTT Broker). When disconnect success, the disconnected event will be triggered.

```
edgeAgent.Disconnect();
```

3.5. UploadConfig(ActionType action, EdgeConfig edgeConfig)

Upload SCADA/Device/Tag Config with Action Type (Create/Update/Delete).

```
EdgeConfig config = new EdgeConfig();
// set scada condig
// set tag config
bool result = _edgeAgent.UploadConfig( ActionType.Create, config ).Result;
```

SCADA Config:

```
config.Scada = new EdgeConfig.ScadaConfig()
{
    Id = textBoxGroupId.Text.Trim(),
    Description = "descrp",
    HeartBeat = 60,
    BackupDeviceId = 0
};
```

Analog Tag Config:

```
EdgeConfig.AnalogTagConfig analogTag = new EdgeConfig.AnalogTagConfig()
{
    Name = "ATag01",
    Description = "ATag descrp",
    ReadOnly = false,
    ArraySize = 0,
    NeedLog = true,
```

```
    SpanHigh = 1000,
    SpanLow = 0,
    EngineerUnit = string.Empty,
    DisplayFormat = "4.2",
    //...
};
```

Discrete Tag Config:

```
EdgeConfig.DiscreteTagConfig discreteTag = new
EdgeConfig.DiscreteTagConfig()
{
    Name = "DTag01",
    Description = "DTag descrp",
    ReadOnly = false,
    ArraySize = 0,
    AlarmEnable = false,
    State0 = "0",
    State1 = "1",
    State2 = string.Empty,
    State3 = string.Empty,
    State4 = string.Empty,
    State5 = string.Empty,
    State6 = string.Empty,
    State7 = string.Empty,
    //...
};
```

Text Tag Config:

```
EdgeConfig.TextTagConfig textTag = new EdgeConfig.TextTagConfig()
{
   Name = "Text",
```

```
    Description = "Text",
    ReadOnly = false,
    ArraySize = 0
};
```

3.6. SendData(EdgeData data)

Send tag value to cloud.

```
Random random = new Random();
EdgeData data = new EdgeData();
for (int j = 1; j <= numATagCount.Value; j++)
{
    EdgeData.Tag aTag = new EdgeData.Tag()
    {
        DeviceId = textBoxGroupId.Text,
        TagName = "ATag" + j,
        Value = random.Next(100)
    };
    data.TagList.Add(aTag);
}
for (int j = 1; j <= numDTagCount.Value; j++)
{
  EdgeData.Tag dTag = new EdgeData.Tag()
  {
        DeviceId = textBoxGroupId.Text,
        TagName = "DTag" + j,
        Value = j % 2
  };
  data.TagList.Add(dTag);
}
for (int j = 1; j <= numTTagCount.Value; j++)
{
```

```
    EdgeData.Tag tTag = new EdgeData.Tag()
    {
        DeviceId = textBoxGroupId.Text,
        TagName = "TTag" + j,
        Value = "TEST " + j.ToString()
    };
    data.TagList.Add(tTag);
}
data.Timestamp = DateTime.Now;

bool result = edgeAgent.SendData( data ).Result;
```

An array tag value have to use Dictionary<string, T>, T is defined according to the tag type (Analog: double, Discrete: int, Text: string).

```
// analog array tag
Dictionary<string, double> dicVal = new Dictionary<string, double>();
dicVal.Add( "0", 0.5 );    // tag index is 0, and it's value is 0.5
dicVal.Add( "1", 1.42 );    // tag index is 1, and it's value is 1.42
dicVal.Add( "2", 2.89 );    // tag index is 2, and it's value is 2.89
EdgeData.Tag aTag = new EdgeData.Tag()
{
   TagName = "ATag",
   Value = dicVal
};
// discrete array tag
Dictionary<string, int> dicVal = new Dictionary<string, int>();
dicVal.Add( "0", 0 );    // tag index is 0, and it's value is 0
dicVal.Add( "1", 1 );    // tag index is 1, and it's value is 1
dicVal.Add( "2", 2 );    // tag index is 2, and it's value is 2
EdgeData.Tag dTag = new EdgeData.Tag()
{
```

```
    TagName = "DTag",
    Value = dicVal
};
// text array tag
Dictionary<string, string> dicVal = new Dictionary<string, string>();
dicVal.Add( "0", "zero" );    // tag index is 0, and it's value is
"zero"
dicVal.Add( "1", "one" );    // tag index is 1, and it's value is
"one"
dicVal.Add( "2", "two" );    // tag index is 2, and it's value is
"two"
EdgeData.Tag tTag = new EdgeData.Tag()
{
    TagName = "TTag",
    Value = dicVal
};
```

3.7. Property

Property Name	Data Type	Description
IsConnected	boolean	Connection status (read only)