https://www.halvorsen.blog

OPC UA with Visual Studio and C#

Using “OPC UA Server Simulator” and “OPC UA .NET SDK”

Hans-Petter Halvorsen

Contents

- Introduction to OPC
  - Communication Protocol for Data Exchange between Devices from different Manufactures typically used in Industrial and Automation Systems
- OPC UA
  - The Next Generation OPC. Cross-platform. Works with IoT/IIoT
- OPC UA Server Simulator
  - Free OPC UA Server that can be used for Testing and Education
- “OPC UA Client” Tool
  - Free OPC UA Client that can be used for Testing and Education
- OPC UA .NET SDK
  - Free Evaluation license which can be used unlimited for each application but runs only for 30 minutes before restart is required
  - Visual Studio/C# Example
  - Improved Example

OPC

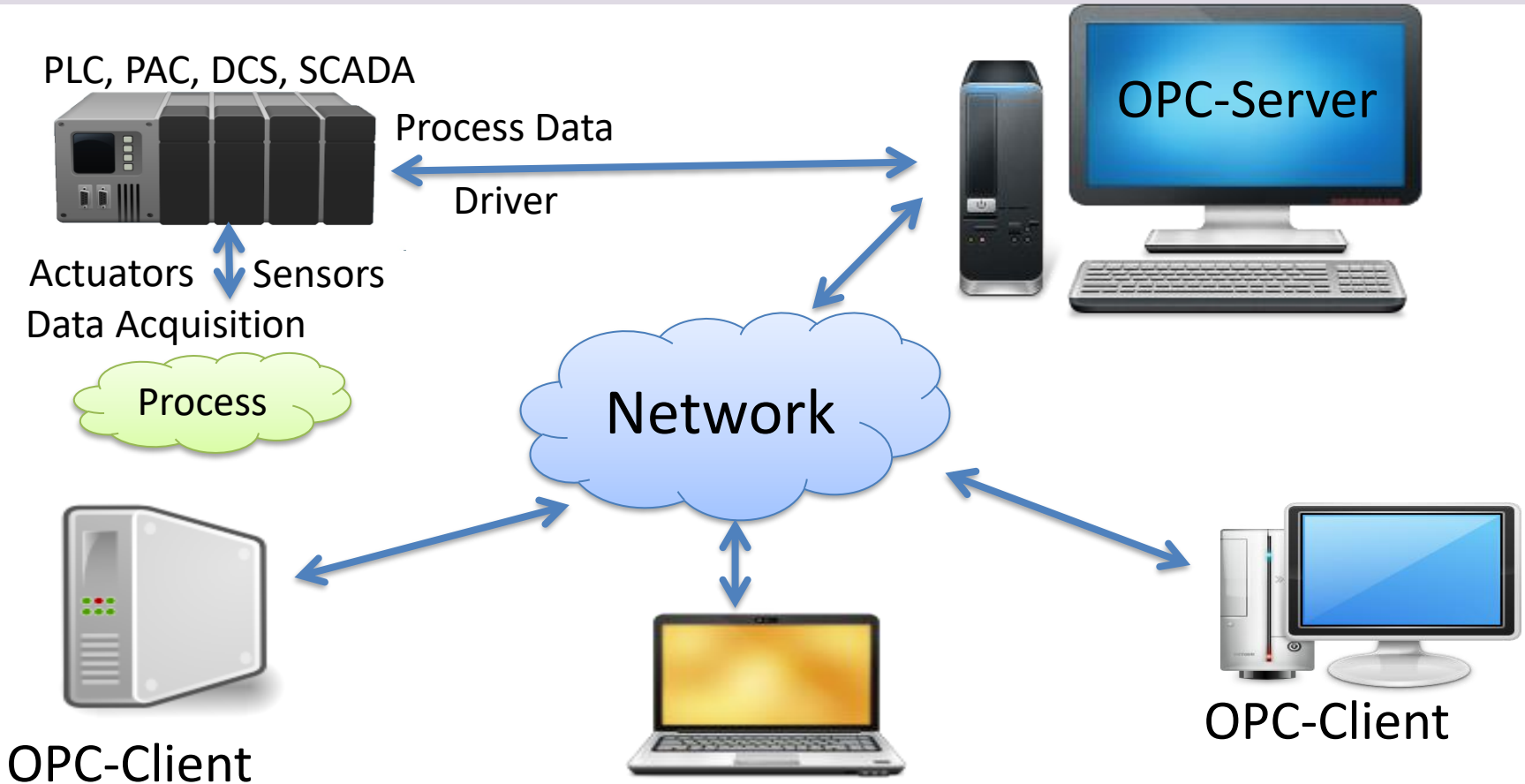
Hans-Petter Halvorsen

Table of Contents

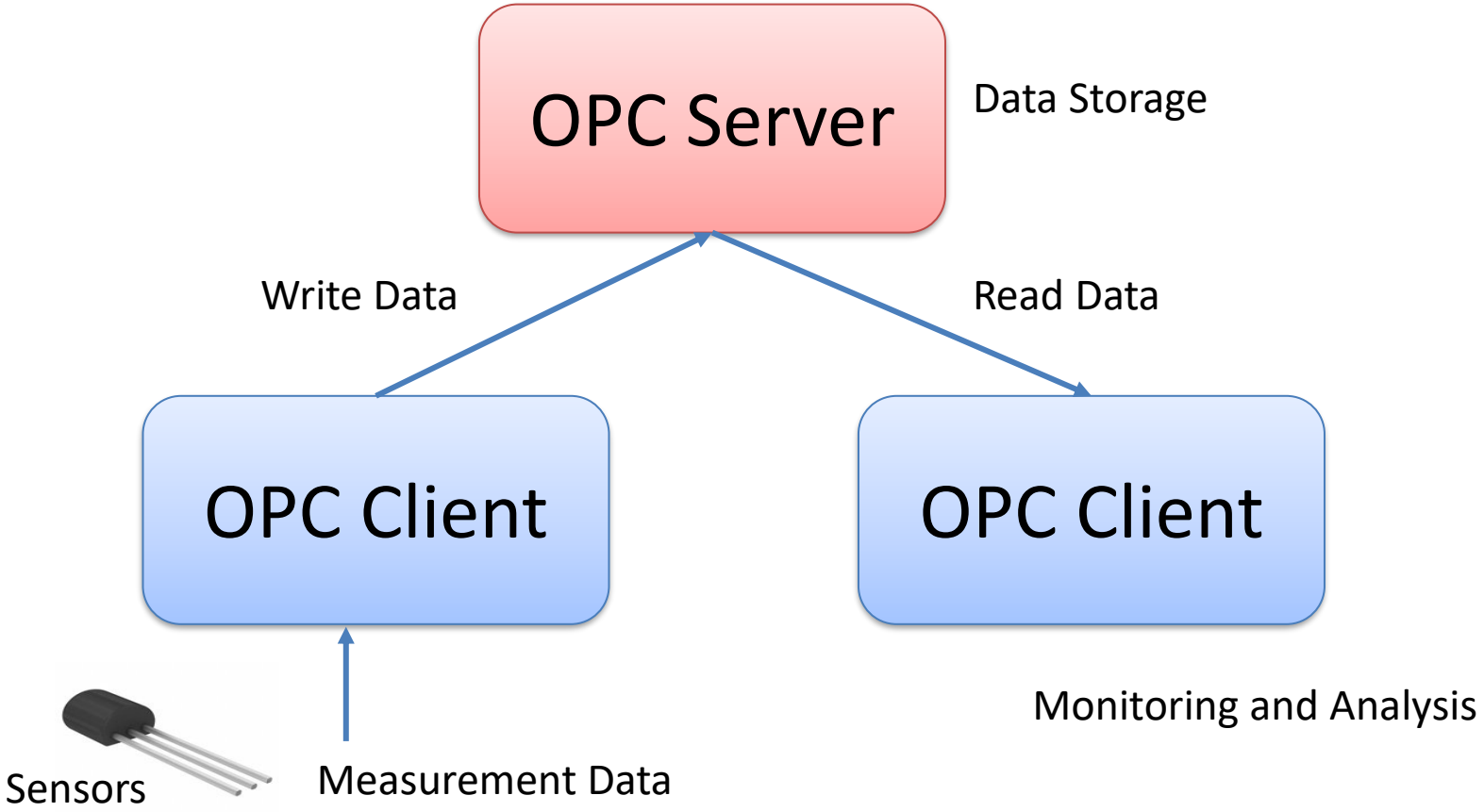
What is OPC?

- A standard that defines the communication of data between devices from different manufactures
- Requires an OPC server that communicates with the OPC clients
- OPC allows “plug-and-play”, gives benefits as reduces installation time and the opportunity to choose products from different manufactures
- Different standards: “Real-time” data (OPC DA), Historical data (OPC HDA), Alarm & Event data (OPC AE), etc.

Typical OPC Scenario



OPC Server and Client(s)



OPC UA

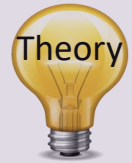
Hans-Petter Halvorsen

Table of Contents

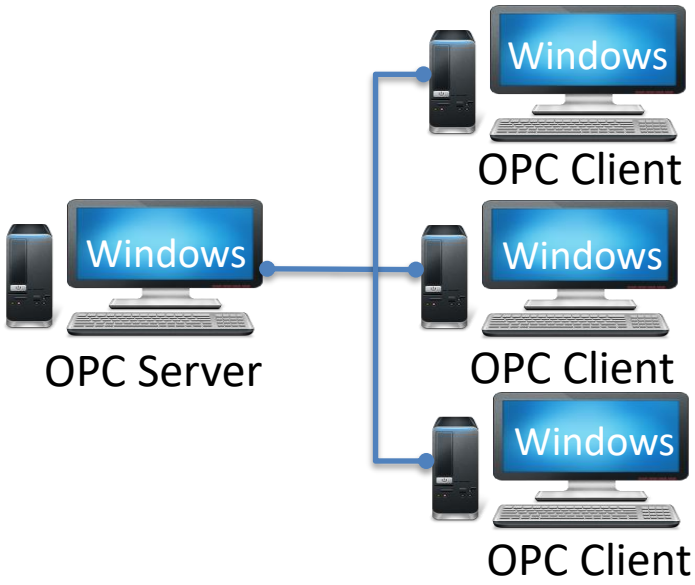
OPC UA

- UA – Unified Architecture
- The Next Generation OPC
- Cross Platform. “Classic“ OPC works only for Windows
- Based on Modern Software/Network Architecture (No DCOM problems!)
- It makes it easier to transmit and receive data in a modern data network/Internet

Classic OPC vs. OPC UA

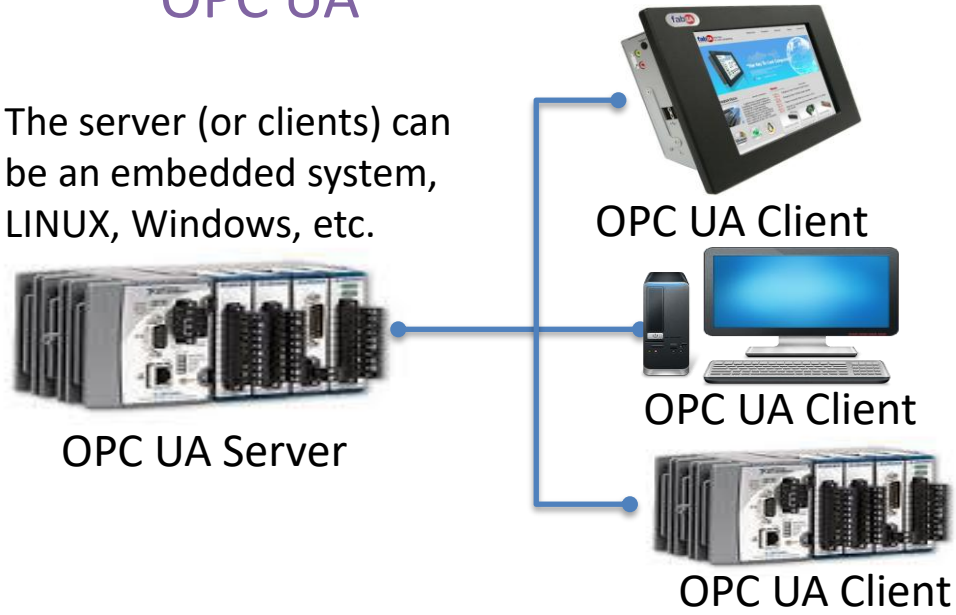


Classic OPC (DCOM)



OPC UA

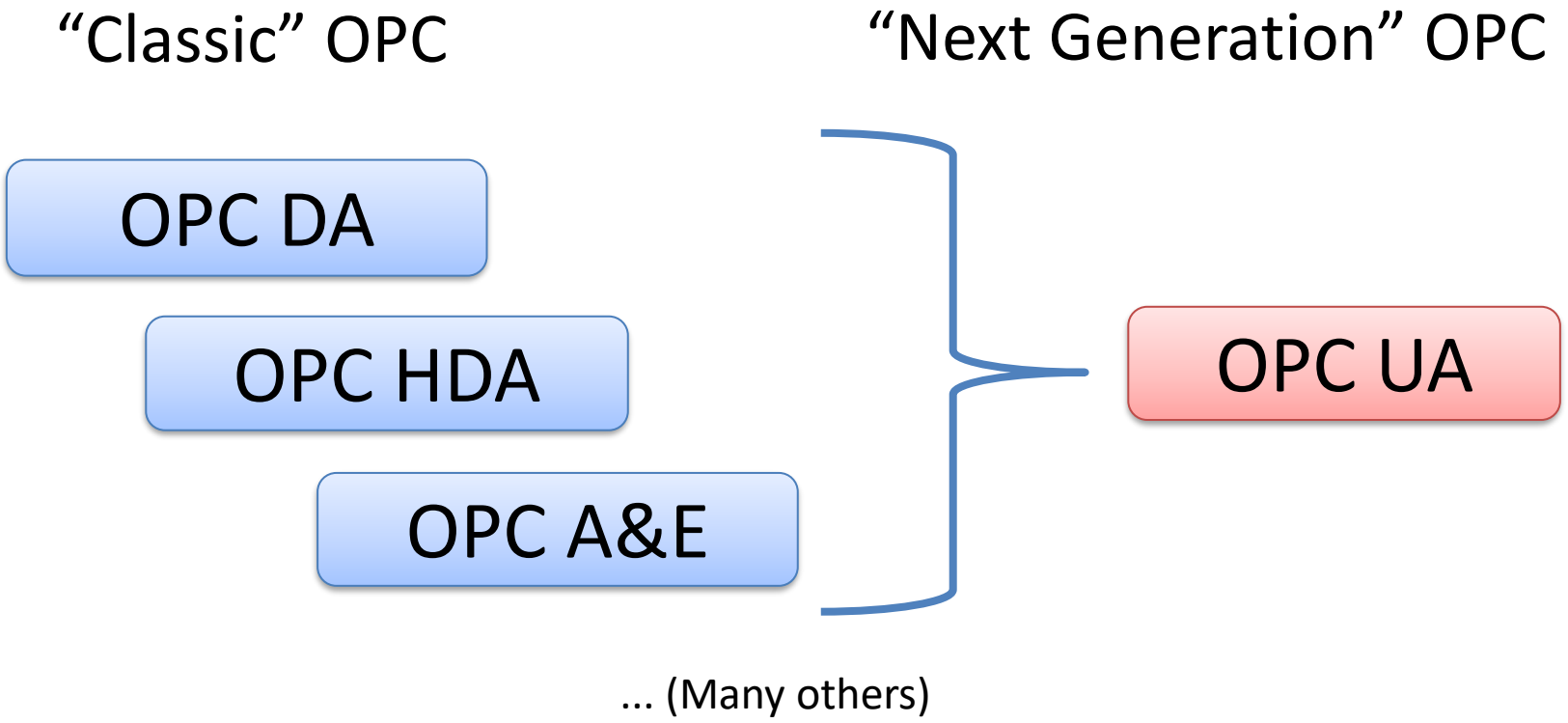
The server (or clients) can be an embedded system, LINUX, Windows, etc.



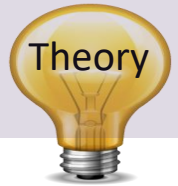
Classic OPC requires a Microsoft Windows operating system to implement COM/DCOM server functionality. By utilizing SOA and Web Services, OPC UA is a platform-independent system that eliminates the previous dependency on a Windows operating system. By utilizing SOAP/XML over HTTP, OPC UA can deploy on a variety of embedded systems regardless of whether the system is a general purpose operating system, such as Windows, or a deterministic real-time operating system.

http://www.ni.com/white-paper/13843/en/

OPC Specifications



Next Generation OPC



COM/DCOM

OPC Classic

OPC DA

OPC HDA

OPC A&E

Specifications

Windows only

Next Generation OPC

XML, HTTP, SOAP

OPC UA

Cross-platform

Windows, Linux, Mac, Embedded, VxWorks

All specifications collected in one (DA, HDA, A&E)

Protocols: “UA Binary” or “UA XML”

OPC DA Server

OPC HDA Server

OPC A&E Server

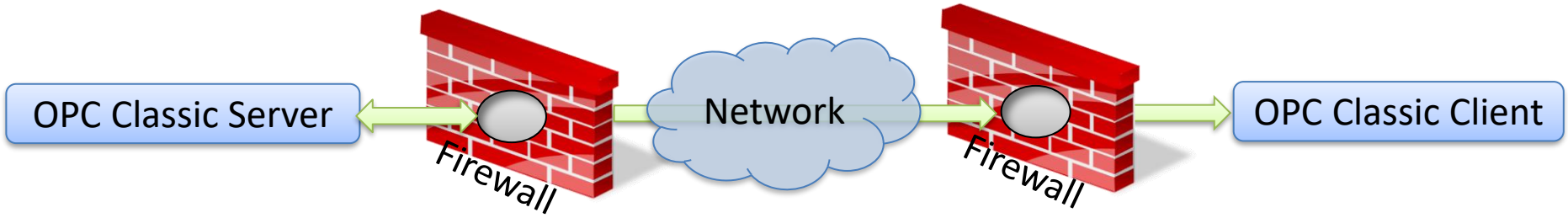
OPC Client (DCOM)

Simpler!!

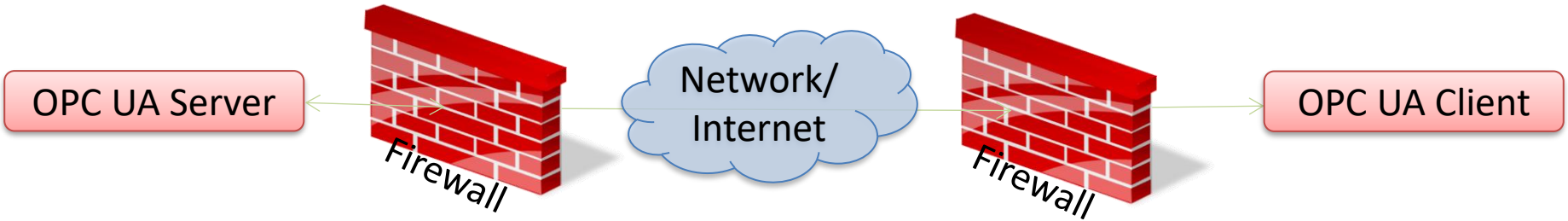
OPC UA Server

OPC UA Client

Next Generation OPC



To open DCOM through firewalls demanded a large hole in the firewall!
Impossible to route over Internet!



No hole in firewall (UA XML) or just a simple needle stick (UA Binary) is necessary
Easy to route over Internet!

OPC UA Server Simulator

Hans-Petter Halvorsen

Table of Contents

OPC UA Server Simulator

- This free OPC UA Server tool supports data access and historical access information models of OPC UA.
- Consequently, it provides simulated real-time and historical data.
- It is possible to configure your own tags and the data simulation via CSV files.
- OPC UA clients can monitor real-time data and explore history data from this simulator.
- https://opcfoundation.org/products/view/opc-ua-server-simulator

OPC UA Server Simulator

https://opcfoundation.org

https://opcfoundation.org/products/view/opc-ua-server-simulator

OPC Tunneling
OPC UA
OPC UA Server Simulator – Full Edition
OPC UA Server Toolkit
OPC UA IoT Broker
OPC UA Server for Databases
OPC UA Client Toolkit
OPC UA Server Simulator
OPC UA Proxy
OPC UA Wrapper
OPC UA Client
OPC Data Archiving
OPC Clients
OPC Servers
OPC Client Toolkits
OPC Free Tools
OPC Server Toolkits

OPC UA Server Simulator

Download | User Guide

Watch Demo Videos

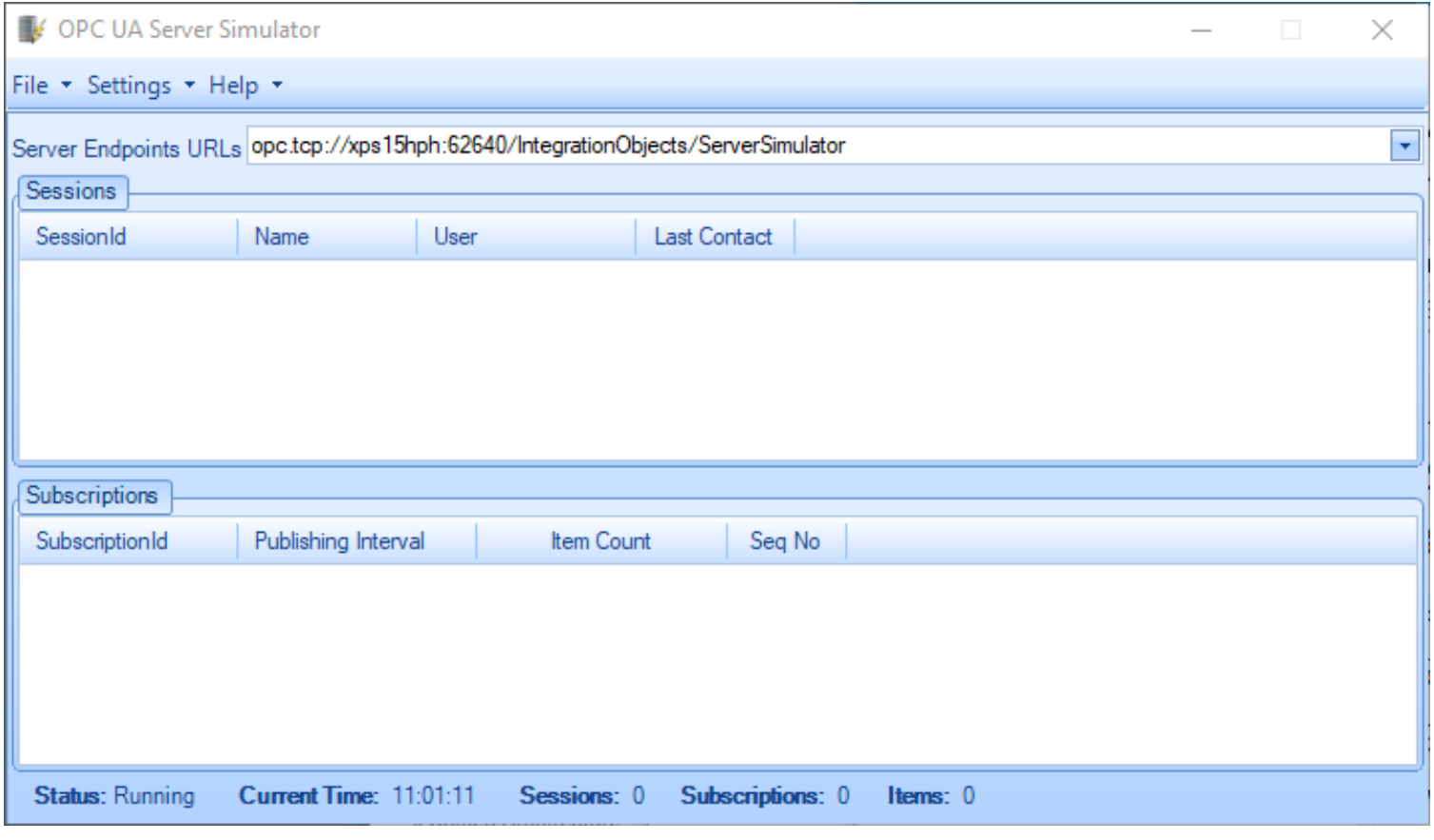
Simulate real-time and historical data using OPC UA Server Simulator!

Integration Objects' **OPC UA Server Simulator** is a free to use and distribute OPC Unified Architecture server utility. Indeed, you can use this OPC UA simulator to play the role of OPC UA servers and test your OPC UA Client applications.

This free OPC UA Server tool supports data access and historical access information models of OPC UA. Consequently, it provides simulated real-time and historical data. Moreover, users can configure their own tags and the data simulation via CSV files. OPC UA clients can monitor real-time data and explore history data from this simulator.

https://integrationobjects.com/sioth-opc/sioth-opc-unified-architecture/opc-ua-server-simulator/

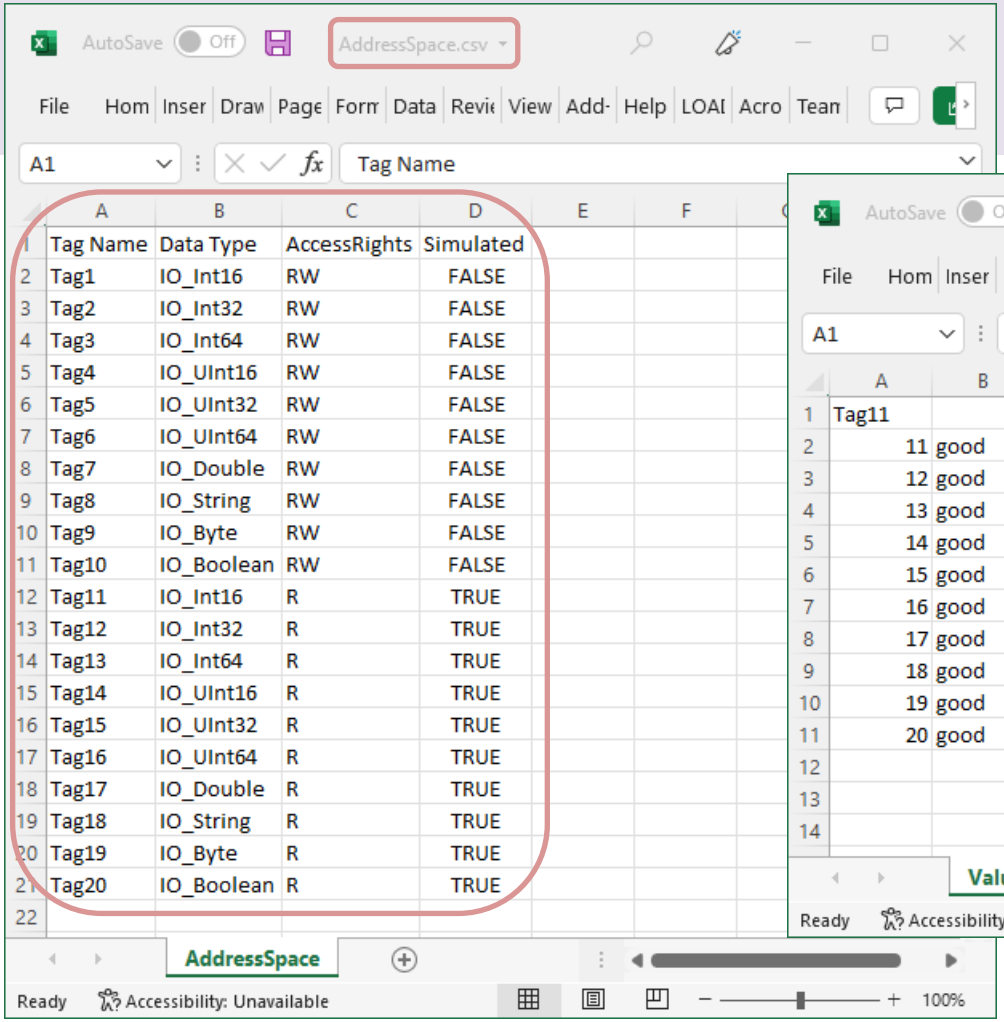
OPC UA Server Simulator



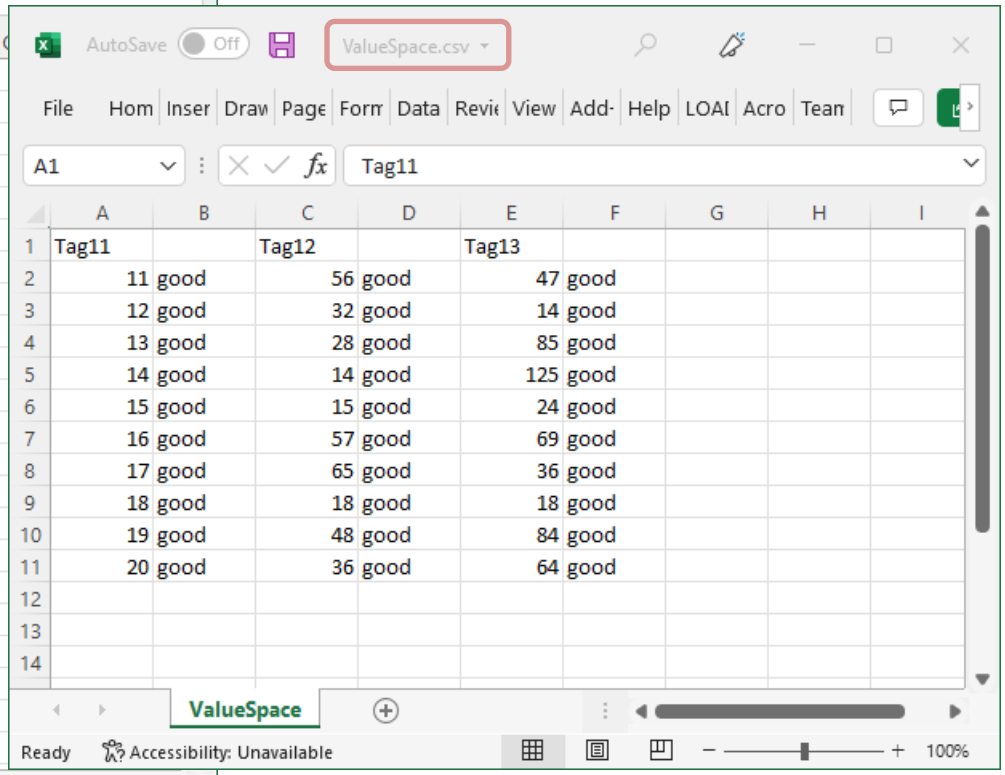
OPC UA Server Simulator

The OPC UA Server Simulator uses 2 CSV simulation files:

- "**AddressSpace.csv**" used to build the address space of the OPC UA Server.
- "**ValueSpace.csv**" used to simulate the data values of the OPC UA items.
- Those two files are located at the following path: X:\Program Files (x86)\Integration Objects\Integration Objects' OPC UA Server Simulator\OPC UA Server Simulator\DATA



	A	B	C	D
1	Tag Name	Data Type	AccessRights	Simulated
2	Tag1	IO_Int16	RW	FALSE
3	Tag2	IO_Int32	RW	FALSE
4	Tag3	IO_Int64	RW	FALSE
5	Tag4	IO_UInt16	RW	FALSE
6	Tag5	IO_UInt32	RW	FALSE
7	Tag6	IO_UInt64	RW	FALSE
8	Tag7	IO_Double	RW	FALSE
9	Tag8	IO_String	RW	FALSE
10	Tag9	IO_Byte	RW	FALSE
11	Tag10	IO_Boolean	RW	FALSE
12	Tag11	IO_Int16	R	TRUE
13	Tag12	IO_Int32	R	TRUE
14	Tag13	IO_Int64	R	TRUE
15	Tag14	IO_UInt16	R	TRUE
16	Tag15	IO_UInt32	R	TRUE
17	Tag16	IO_UInt64	R	TRUE
18	Tag17	IO_Double	R	TRUE
19	Tag18	IO_String	R	TRUE
20	Tag19	IO_Byte	R	TRUE
21	Tag20	IO_Boolean	R	TRUE
22				



	A	B	C	D	E	F
1	Tag11		Tag12		Tag13	
2	11	good	56	good	47	good
3	12	good	32	good	14	good
4	13	good	28	good	85	good
5	14	good	14	good	125	good
6	15	good	15	good	24	good
7	16	good	57	good	69	good
8	17	good	65	good	36	good
9	18	good	18	good	18	good
10	19	good	48	good	84	good
11	20	good	36	good	64	good
12						
13						
14						

“OPC UA Client” Tool

Hans-Petter Halvorsen

Table of Contents

“OPC UA Client” Tool

- “OPC UA Client” is a free client tool that supports the main OPC Unified Architecture information models.
- These models are Data Access, Alarms & Conditions, and Historical Data Access
- https://integrationobjects.com/sioth-opc/sioth-opc-unified-architecture/opc-ua-client/

- OPC Tunneling
- OPC UA
  - OPC UA Server Simulator – Full Edition
  - OPC UA Server Toolkit
  - OPC UA IoT Broker
  - OPC UA Server for Databases
  - OPC UA Client Toolkit
  - OPC UA Server Simulator
  - OPC UA Proxy
  - OPC UA Wrapper
  - OPC UA Client
- OPC Data Archiving
- OPC Clients
- OPC Servers
- OPC Client Toolkits
- OPC Free Tools
- OPC Server Toolkits

OPC UA Client

Download | User Guide | Quick User Guide

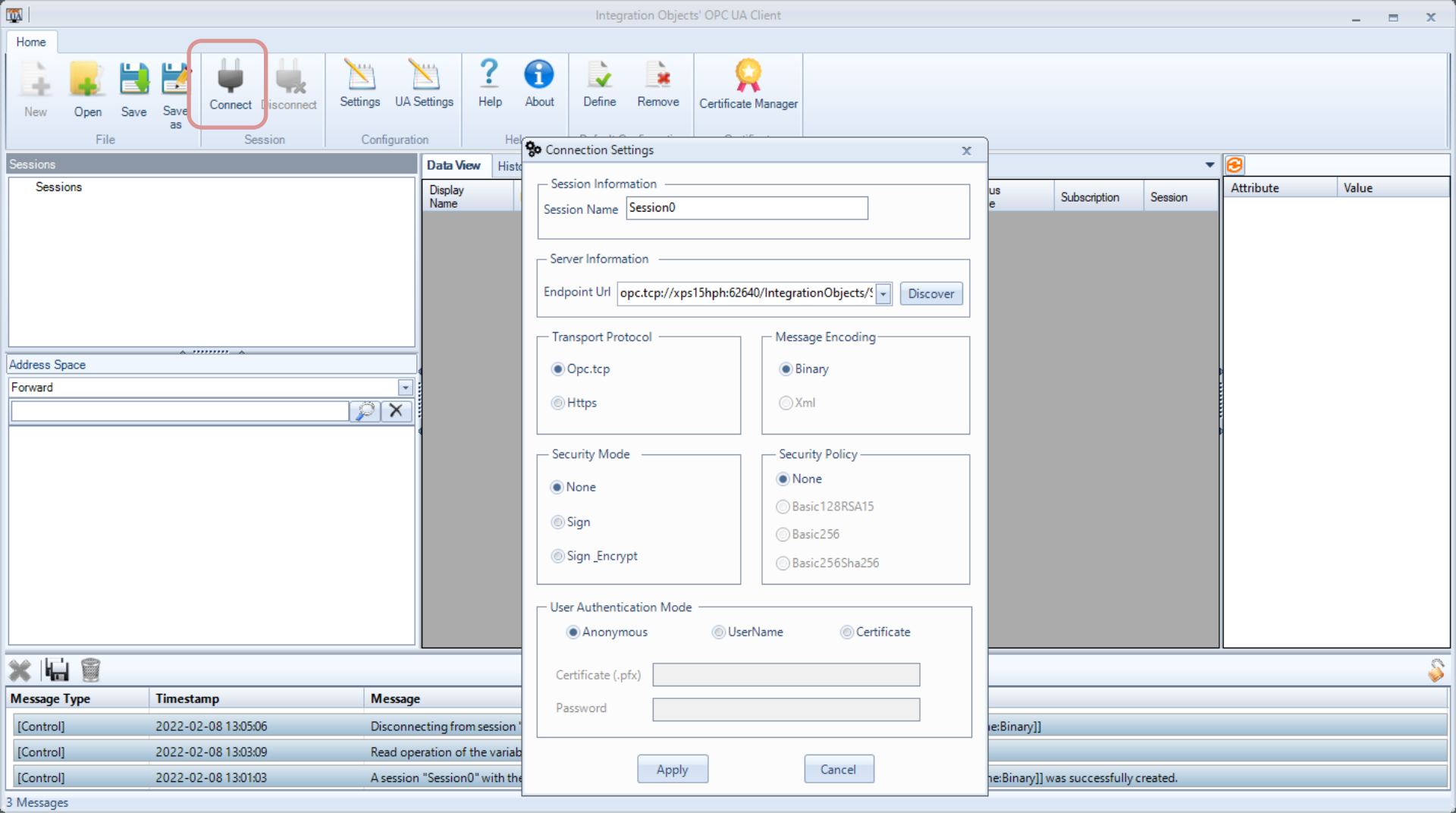
Download free OPC UA Client and start your OPC UA tests now!

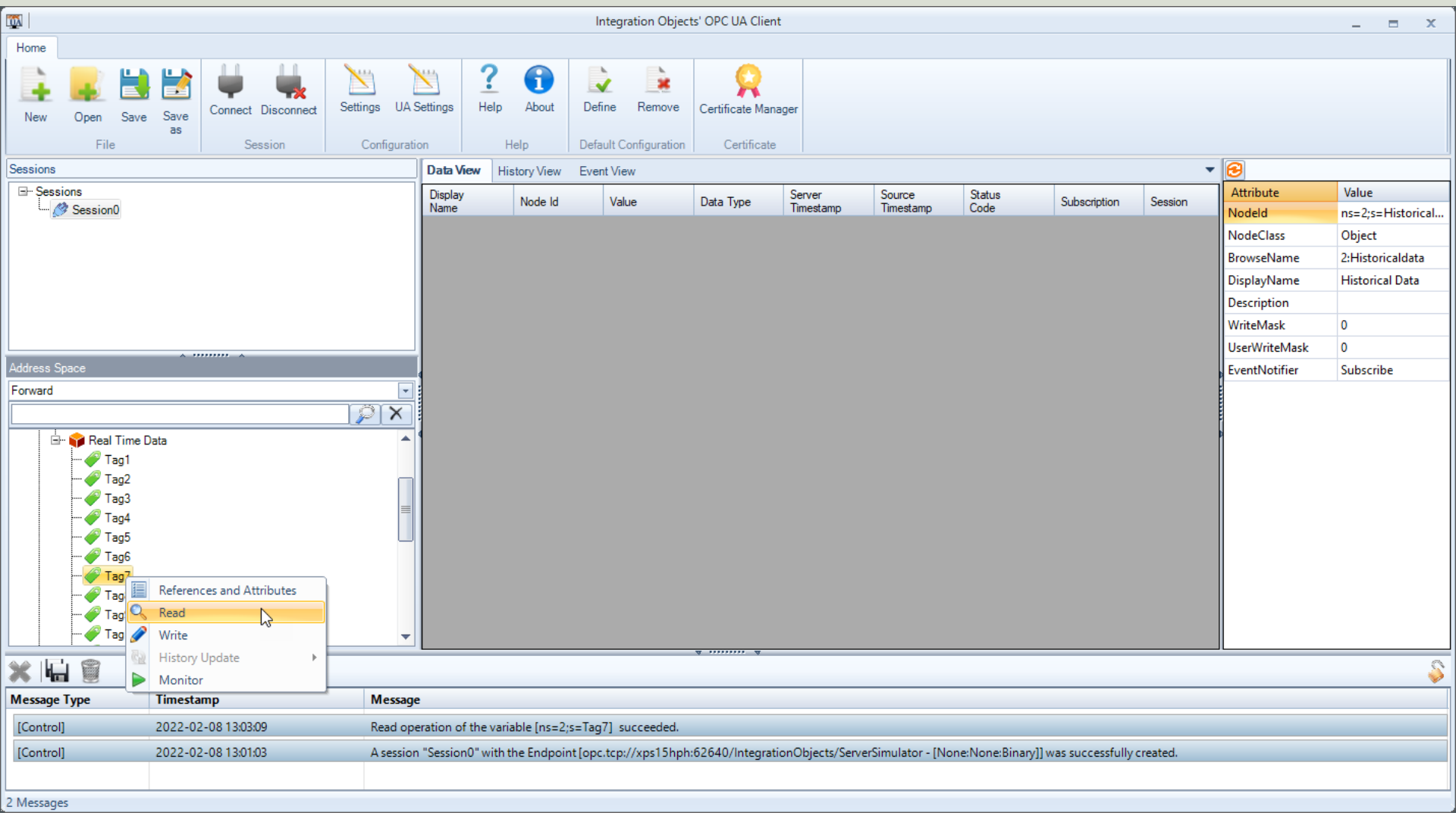
OPC UA Client is a free client tool that supports the main OPC Unified Architecture information models. These models are Data Access, Alarms & Conditions, and Historical Data Access. In fact, it offers the capability to:

- Discover local and remote OPC UA servers
- Establish secure communication channels
- Browse the address space of any OPC UA compliant server
- Monitor real-time data and alarms & conditions
- Explore and update history data

Moreover, this OPC UA explorer allows you to generate its self-signed Application Instance Certificate in order to provide application level security and secure the connections with OPC UA servers.

View Tutorial Video of OPC UA Test Client & OPC UA Wrapper


Integration Objects' OPC UA Client
Home
New
Open
Save
Save as
Connect
Disconnect
Settings
UA Settings
Help
About
Define
Remove
Certificate Manager
File
Session
Configuration
Sessions
Sessions
Address Space
Forward
Data View
Display Name
Subscription
Session
Attribute
Value
Connection Settings
Session Information
Session Name
Session0
Server Information
Endpoint Url
opc.tcp://xps15hph:62640/IntegrationObjects/S
Discover
Transport Protocol
Opc.tcp
Https
Message Encoding
Binary
Xml
Security Mode
None
Sign
Sign _Encrypt
Security Policy
None
Basic128RSA15
Basic256
Basic256Sha256
User Authentication Mode
Anonymous
UserName
Certificate
Certificate (.pfx)
Password
Apply
Cancel
Message Type
Timestamp
Message
[Control]
2022-02-08 13:05:06
Disconnecting from session
[Control]
2022-02-08 13:03:09
Read operation of the variab
[Control]
2022-02-08 13:01:03
A session "Session0" with the
:Binary]]
:Binary]] was successfully created.
3 Messages


Integration Objects' OPC UA Client
Home
New
Open
Save
Save as
File
Connect
Disconnect
Session
Settings
UA Settings
Configuration
Help
About
Help
Define
Remove
Default Configuration
Certificate Manager
Certificate
Sessions
Sessions
Session0
Address Space
Forward
Real Time Data
Tag1
Tag2
Tag3
Tag4
Tag5
Tag6
Tag7
References and Attributes
Read
Write
History Update
Monitor
Data View
History View
Event View
Display Name
Node Id
Value
Data Type
Server Timestamp
Source Timestamp
Status Code
Subscription
Session
Attribute
Value
NodeId
ns=2;s=Historical...
NodeClass
Object
BrowseName
2:Historicaldata
DisplayName
Historical Data
Description
WriteMask
0
UserWriteMask
0
EventNotifier
Subscribe
Message Type
Timestamp
Message
[Control]
2022-02-08 13:03:09
Read operation of the variable [ns=2;s=Tag7] succeeded.
[Control]
2022-02-08 13:01:03
A session "Session0" with the Endpoint [opc.tcp://xps15hph:62640/IntegrationObjects/ServerSimulator - [None:None:Binary]] was successfully created.
2 Messages

OPC UA with C#

Hans-Petter Halvorsen

Table of Contents

OPC UA with Visual Studio/C#

- Lots of Packages and Libraries do exist for creating both OPC UA Clients and OPC UA Servers in Visual Studio/C#
- Most of them are payment based
- Many of those can be evaluated for a trial period or used forever with some restrictions
- In this Tutorial, “OPC UA .NET SDK” will be used

OPC UA .NET SDK

Hans-Petter Halvorsen

Table of Contents

OPC UA .NET SDK

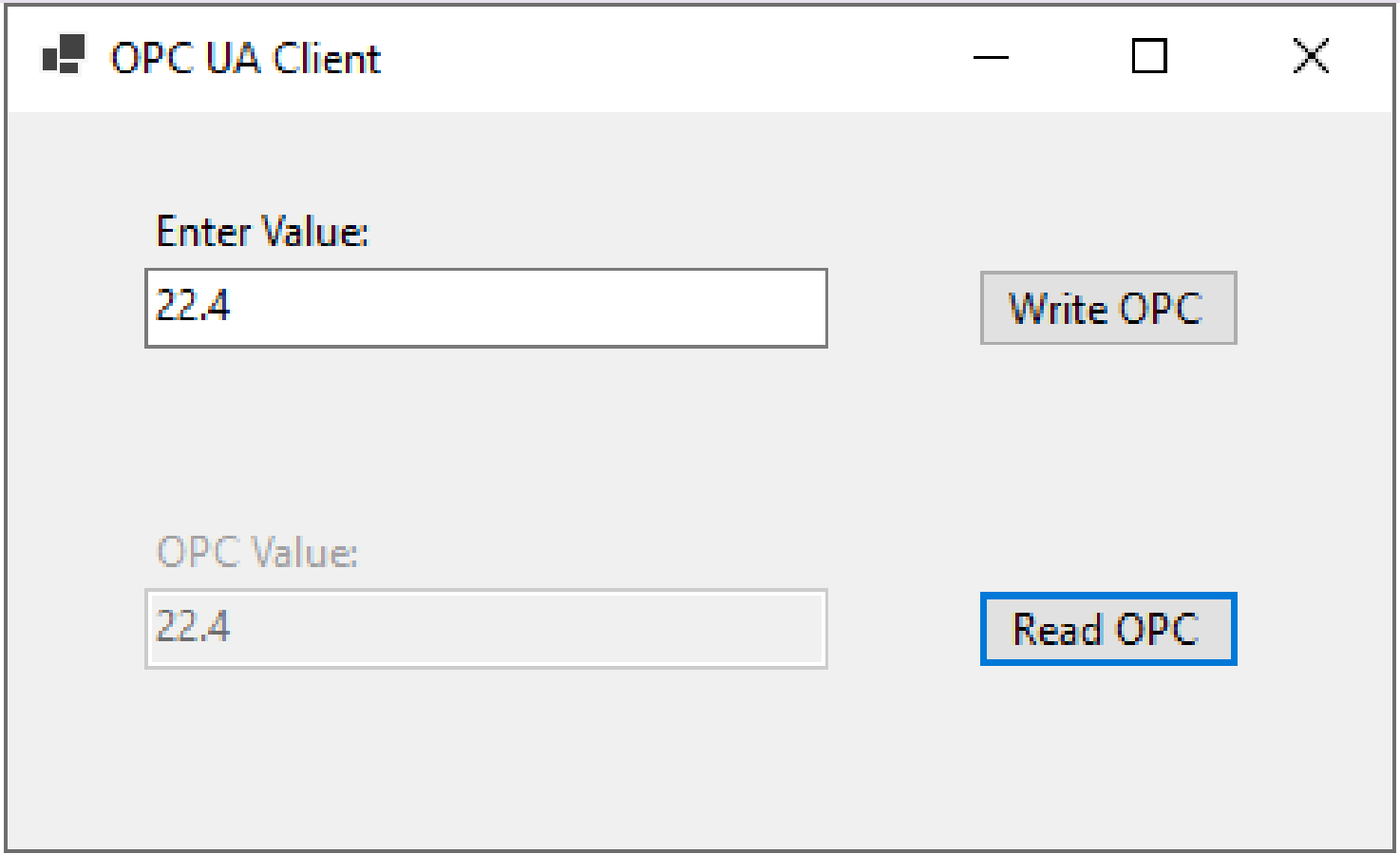
- The “OPC UA .NET SDK” comes with an evaluation license which can be used unlimited for each application run for 30 minutes
- It comes in a NuGet Package you can install and use in your Visual Studio Project
- https://opcfoundation.org/products/view/opc-ua-net-sdk-for-client-and-server

Visual Studio/C# Example

Hans-Petter Halvorsen

Table of Contents

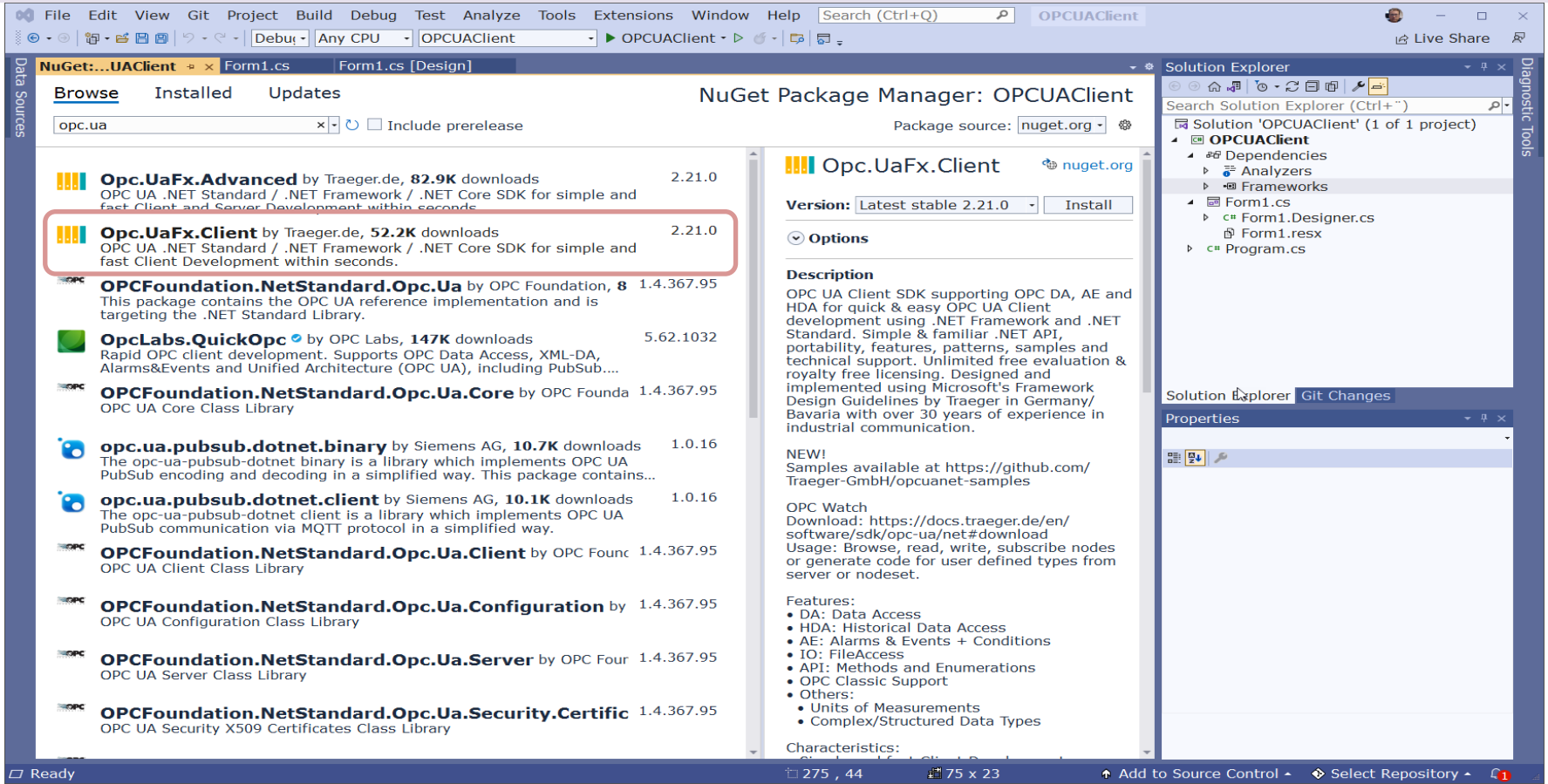
Visual Studio/C# Example



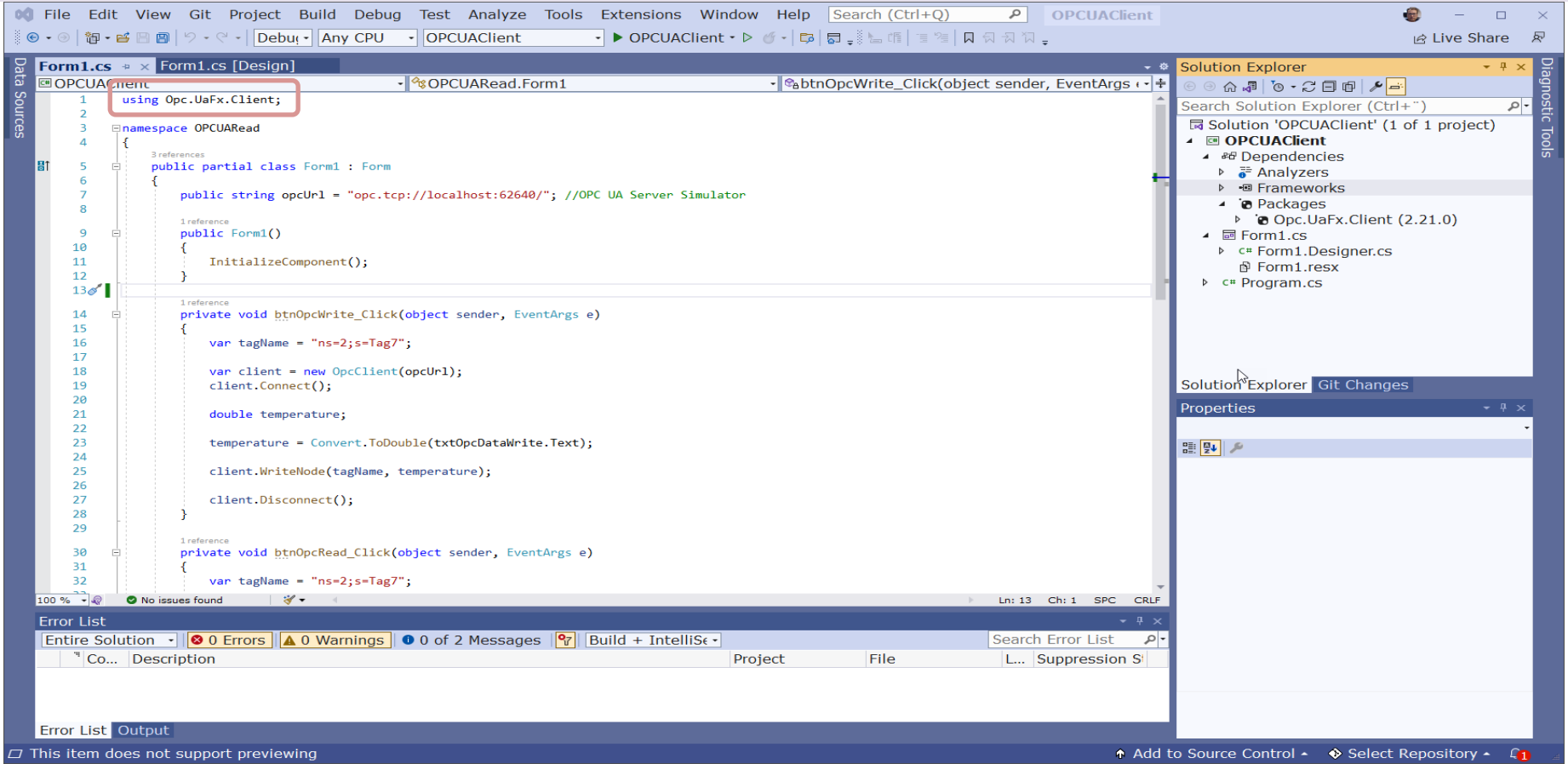
Visual Studio/C# Example

- Note that this is a simplified example in order demonstrate the principle of Writing Data to an OPC UA Server and Reading Data from an OPC Server
- Write and Read the same OPC Tag in the same Application makes no sense in a real scenario
- Typically, the OPC clients are distributed in a network and the different Applications are located on different computers in a network

NuGet Package



Visual Studio Project



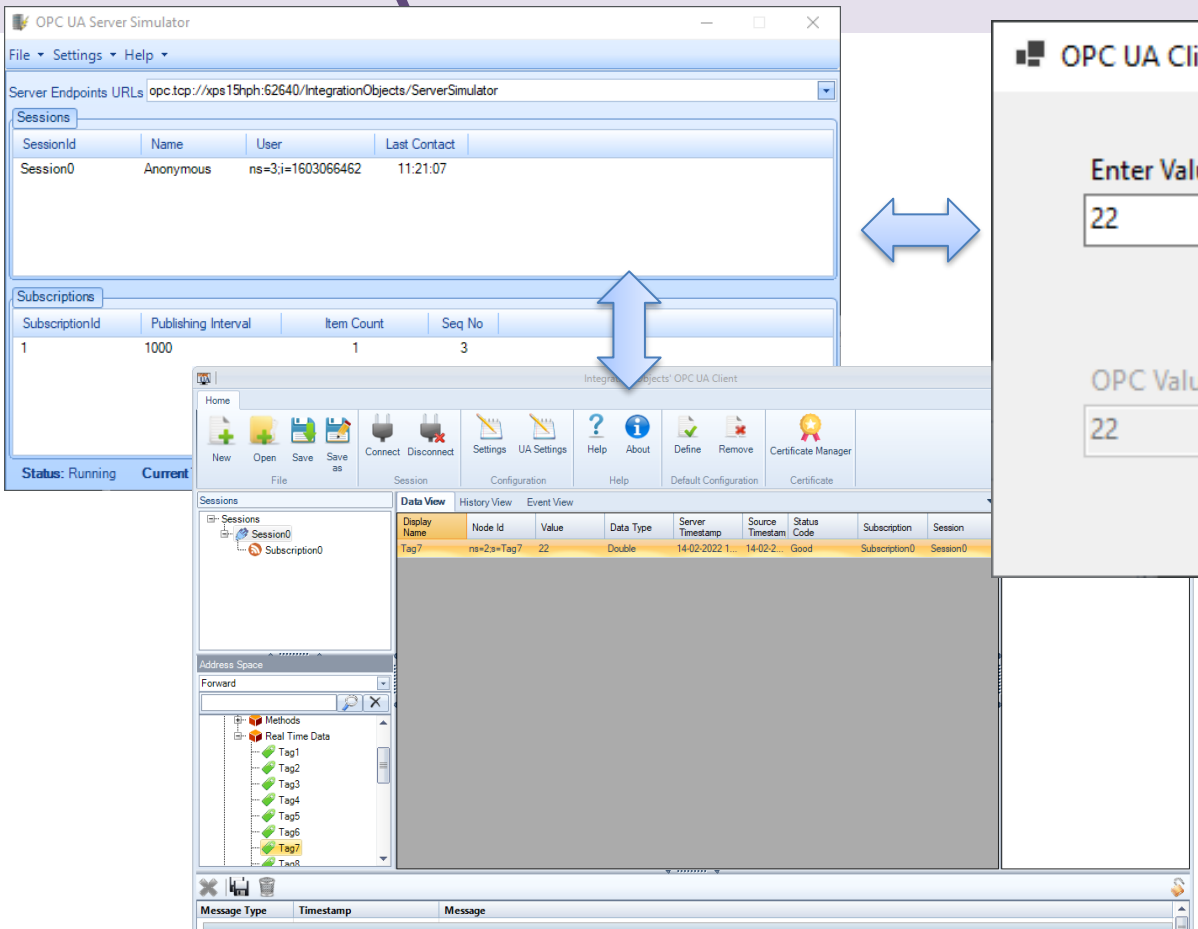
OPC UA Write

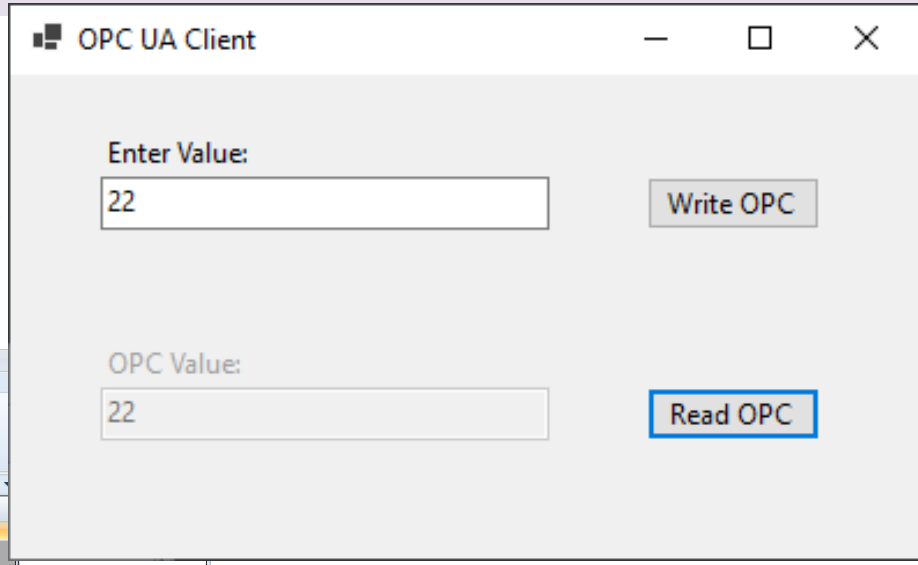
```
private void btnOpcWrite_Click(object sender, EventArgs e)
{
    string opcUrl = "opc.tcp://localhost:62640/";
    var tagName = "ns=2;s=Tag7";

    var client = new OpcClient(opcUrl);
    client.Connect();

    double temperature;
    temperature = Convert.ToDouble(txtOpcDataWrite.Text);

    client.WriteNode(tagName, temperature);

    client.Disconnect();
}
```

OPC UA Read

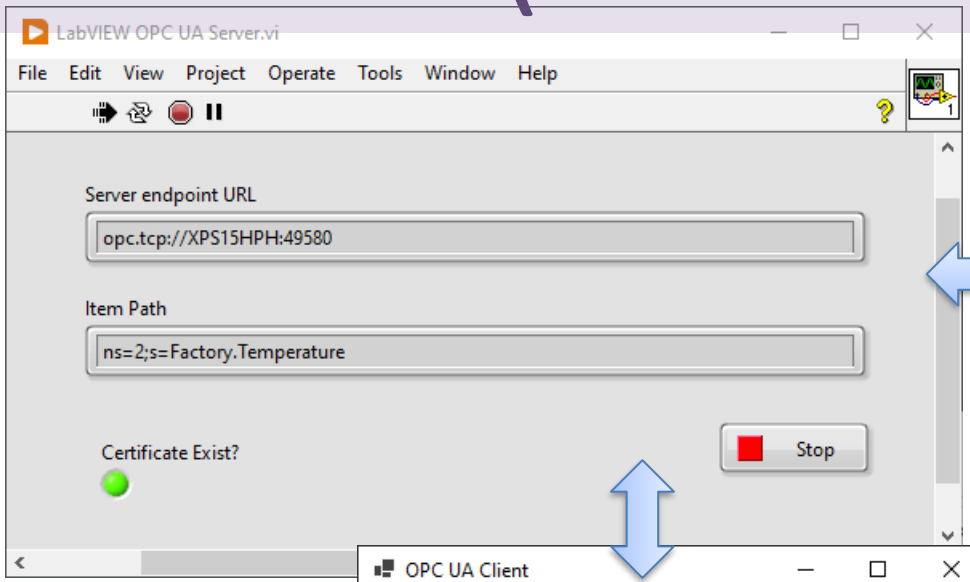
```
private void btnOpcRead_Click(object sender, EventArgs e)
{
    string opcUrl = "opc.tcp://localhost:62640/";
    var tagName = "ns=2;s=Tag7";

    var client = new OpcClient(opcUrl);
    client.Connect();

    var temperature = client.ReadNode(tagName);
    txtOpcDataRead.Text = temperature.ToString();

    client.Disconnect();
}
```

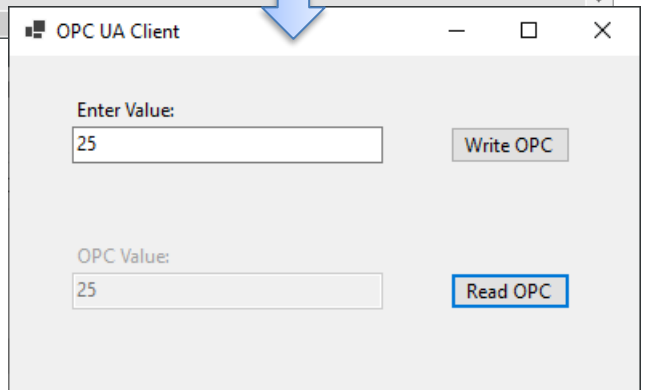
Test 1 (OPC UA Server Simulator)

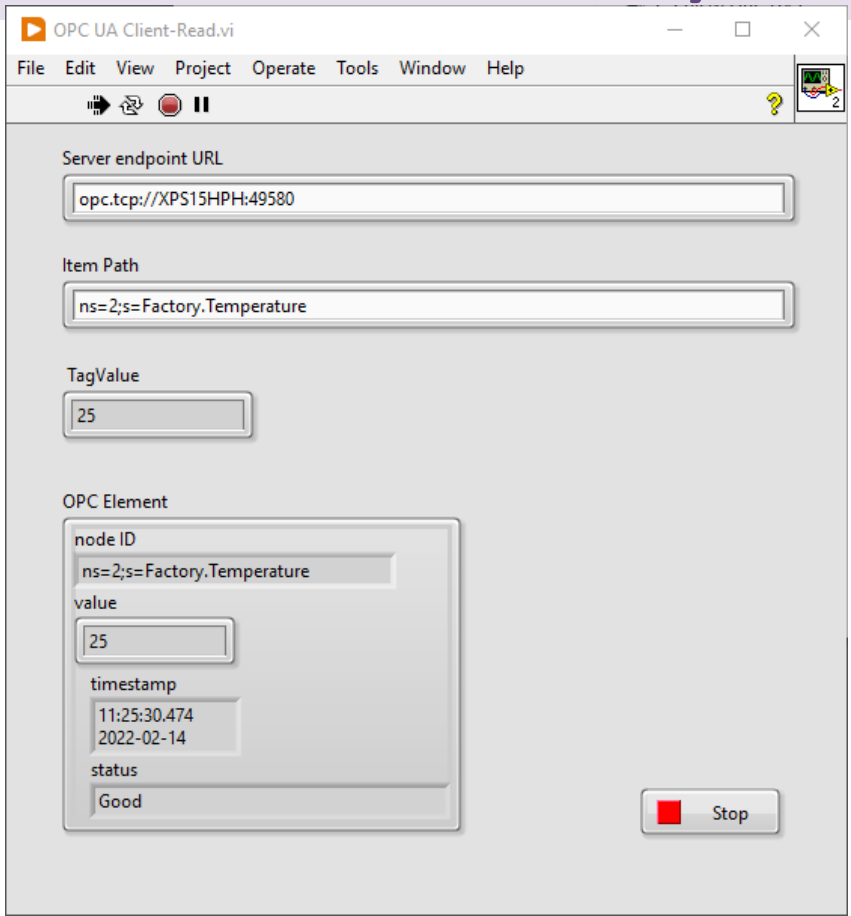




Test 2 (LabVIEW OPC Server)







Improved Example

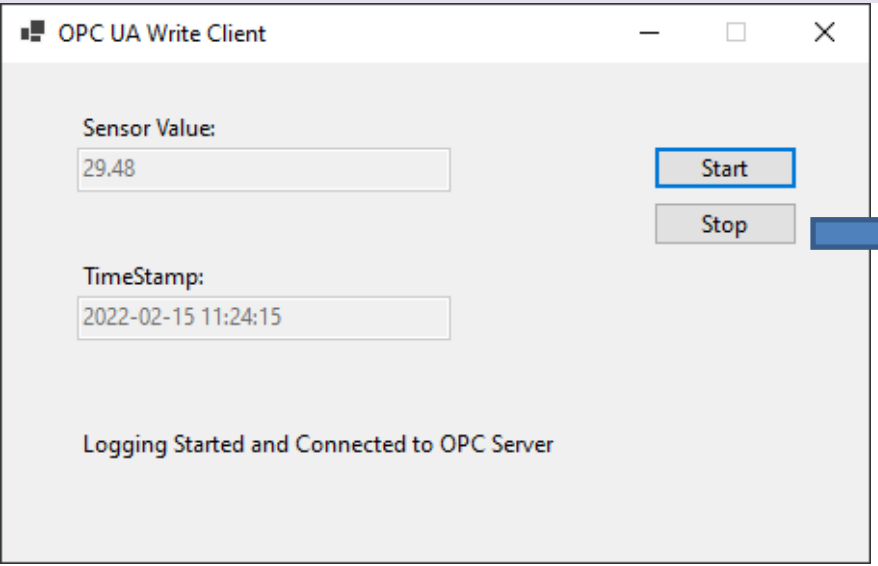
Hans-Petter Halvorsen

Table of Contents

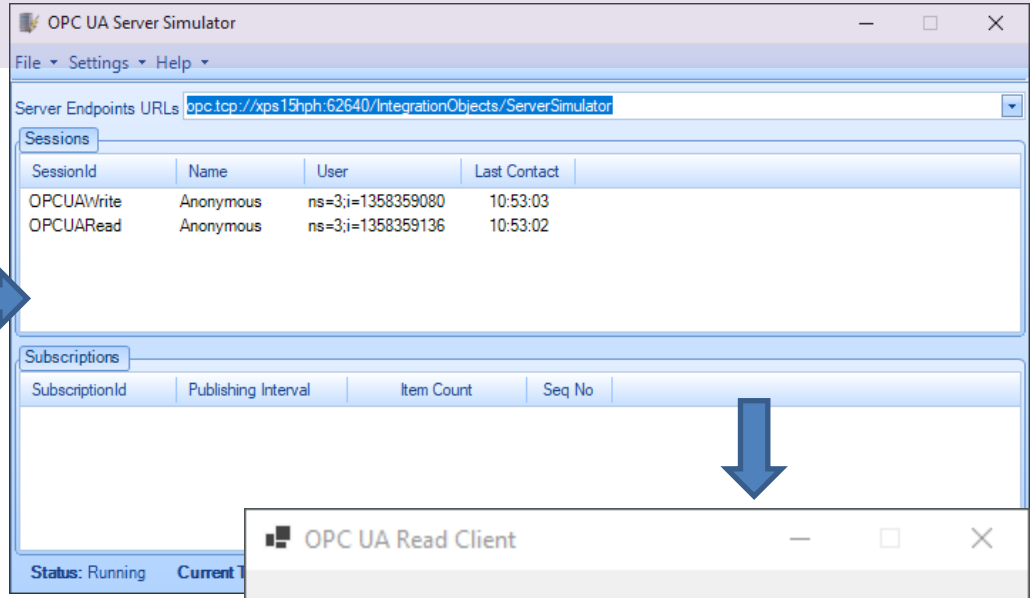
Improved Example

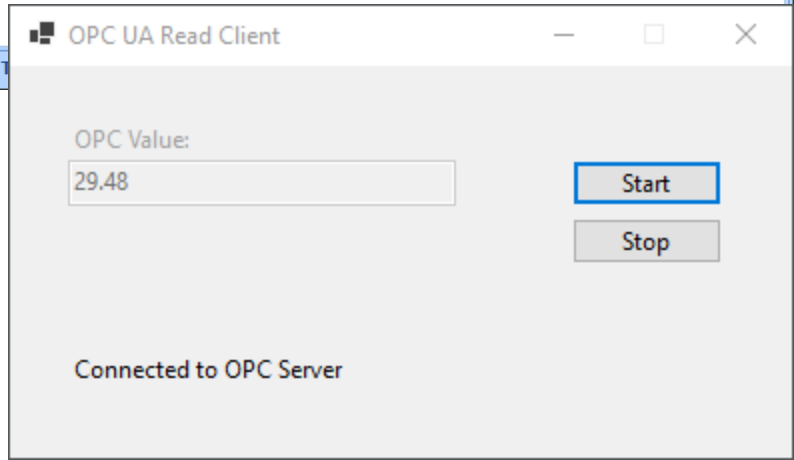
We will implement some Improvements:

- Separate Applications for Write and Read
- Start and Stop Buttons
- Formatting Number of Decimals
- Property window (preventing resizing of the window):
  - form1.FormBorderStyle = FormBorderStyle.FixedSingle;
  - Form1.MaximizeBox = false;
- Using a Timer (Writing/Reading at specific Intervals)
- General Improvements in the Code



OPC UA Write C# App

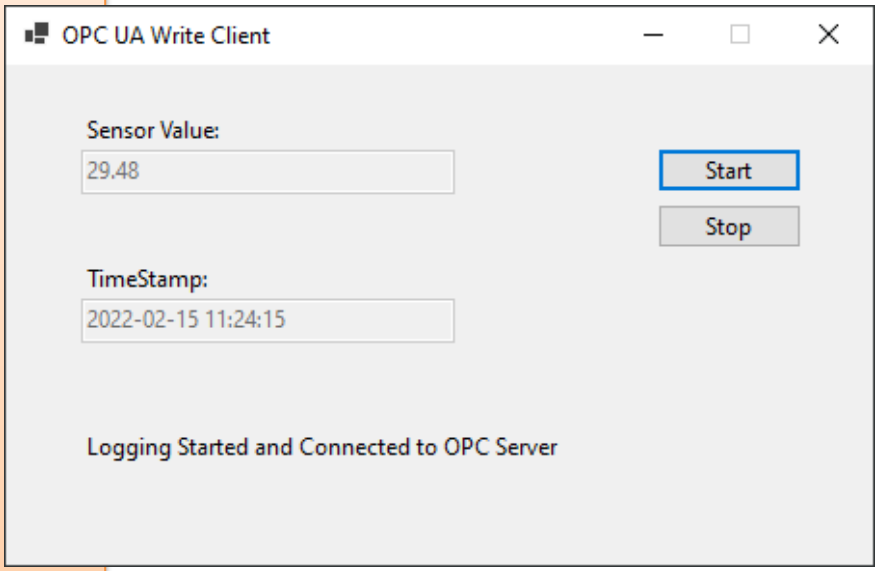




OPC UA Read C# App

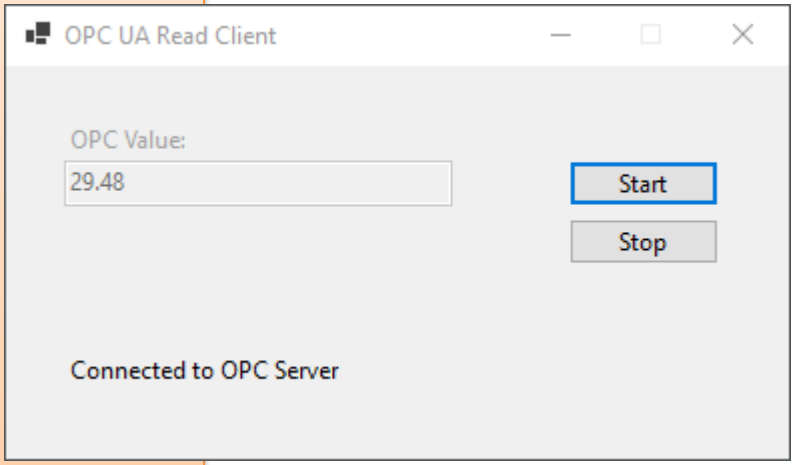
OPC UA Write

```
using Opc.UaFx.Client;

namespace OPCUARead
{
    public partial class Form1 : Form
    {
        public OpcClient client = new OpcClient("opc.tcp://localhost:62640/");

        public Form1()
        {
            InitializeComponent();
            timer1.Interval = 10000;
        }

        private void Form1_FormClosing(object sender, FormClosingEventArgs e)
        {
            if (client != null)
                client.Disconnect();
        }


        private void timer1_Tick(object sender, EventArgs e)
        {
            double sensorValue;

            sensorValue = ReadSensorData();
            OpcWrite(sensorValue);
        }

        double ReadSensorData()
        {
            var rand = new Random();
            int minValue = 20, maxValue = 30;
            double sensorValue;

            sensorValue = rand.NextDouble() * (maxValue - minValue) + minValue;
            txtSensorValue.Text = sensorValue.ToString("#.##");
            DateTime sensorDateTime = DateTime.Now;
            txtTimeStamp.Text = sensorDateTime.ToString("yyyy-MM-dd HH:mm:ss");

            return sensorValue;
        }

        void OpcWrite(double sensorValue)
        {
            string tagName = "ns=2;s=Tag7";
            client.WriteNode(tagName, sensorValue);
        }

        private void btnStart_Click(object sender, EventArgs e)
        {
            client.Connect();
            timer1.Start();
            lblStatusMessage.Text = "Logging Started and Connected to OPC Server";
        }

        private void btnStop_Click(object sender, EventArgs e)
        {
            timer1.Stop();
            if (client != null)
                client.Disconnect();
            lblStatusMessage.Text = "Logging Stopped and Disconnected from OPC Server";
        }

    }
}
```



OPC UA Read

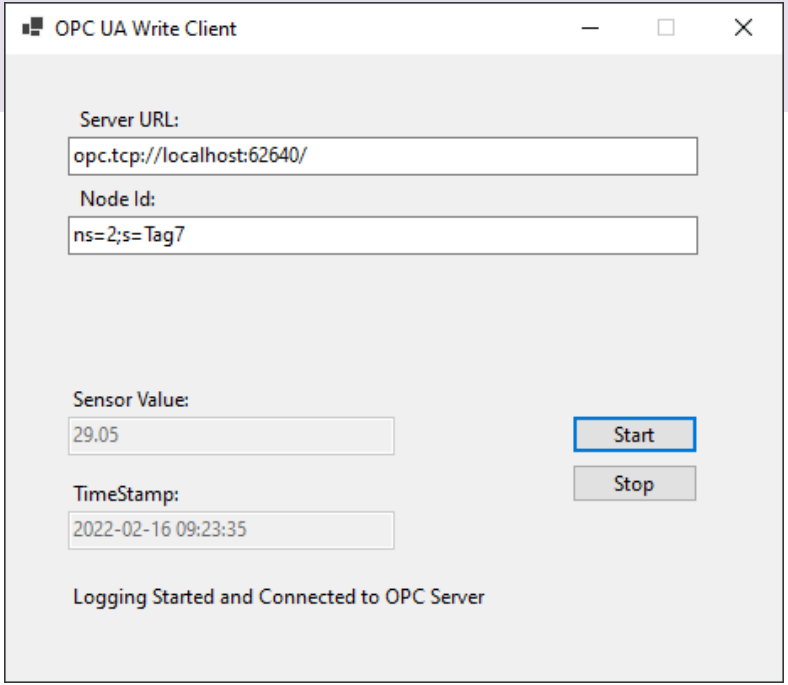
```
using Opc.UaFx.Client;

namespace OPCUARead
{
    public partial class Form1 : Form
    {
        public OpcClient client = new OpcClient("opc.tcp://localhost:62640/");

        public Form1()
        {
            InitializeComponent();
            timer1.Interval = 2000;
        }

        private void timer1_Tick(object sender, EventArgs e)
        {
            OpcRead();
        }

        void OpcRead()
        {
            string tagName = "ns=2;s=Tag7";

            Opc.UaFx.OpcValue opcData = client.ReadNode(tagName);

            double temperature = (double)opcData.Value;

            txtOpcValue.Text = temperature.ToString("#.##");
        }

        private void btnStart_Click(object sender, EventArgs e)
        {
            client.Connect();
            timer1.Start();
            lblStatusMessage.Text = "Connected to OPC Server";

        }

        private void btnStop_Click(object sender, EventArgs e)
        {
            timer1.Stop();
            if (client != null)
                client.Disconnect();
            lblStatusMessage.Text = "Disconnected from OPC Server";
        }

        private void Form1_FormClosing(object sender, FormClosingEventArgs e)
        {
            if (client != null)
                client.Disconnect();
        }

    }
}
```



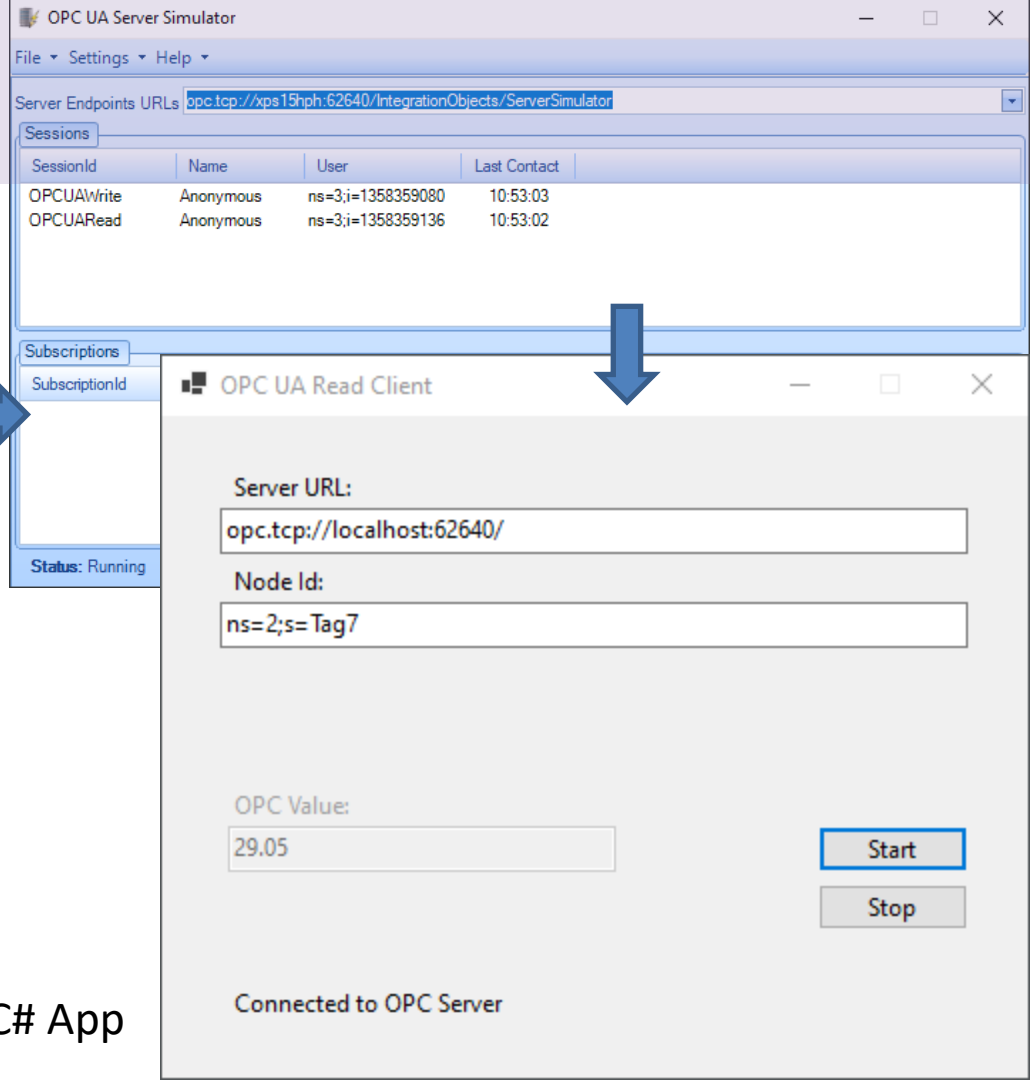
More Improvements

We will implement some more Improvements:

- Possible to specify ServerUrl and NodeId (TagName) from the GUI



OPC UA Write C# App



OPC UA Read C# App

Further Work

Still, lots of improvements can be made

- Possible to change the Logging Interval from the GUI
- Add a Chart
- Get Data from a real Sensor
- Add Units
- Etc.
- In addition, the “OPC UA .NET SDK” comes with many features that will not be demonstrated here

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: https://www.halvorsen.blog